WEST Search History

DATE: Wednesday, July 30, 2003

Set Name side by side	Query	Hit Count	Set Name result set
	DB=USPT,PGPB; PLUR=YES; OP=ADJ		
L21	L19 and ((wake adj2 up) or wake-up)	3	L21
L20	L19 and (wake adj2 up)	3	L20
L19	l17 and PSTN	44	L19
L18	l11 same l1	0	L18
L17	communication near8 (remote adj4 access) near8 (device or proxy)	237	L17
L16	l11 same l1	0	L16
L15	(secure adj4 agent) same (remote adj4 access)	0	L15
L14	(secure adj4 agent) same (remote adj4 access) same (SILI adj4 server)	0	L14
L13	L12 and l2	5	L13
L12	L11 and l1	57	L12
L11	secure adj4 agent	295	L11
L10	L9 and (Internet or (service adj5 provider))	39	L10
L9	L7 and (connection near8 request)	41	L9
L8	L7 and (wake adj2 up)	0	L8
L7	L6 and (register or registration)	58	L7
L6	L4 and ((base adj4 device) or proxy)	95	L6
L5	L4 and (wake-up)	0	L5
L4	l2 and (server adj5 communication)	234	L4
L3	l2 and (job adj4 handler)	0	L3
L2	L1 and (remote adj4 user)	638	L2
L1	remote adj4 access adj4 (device or system)	1770	L1

END OF SEARCH HISTORY

L10: Entry 8 of 39 File: PGPB Mar 21, 2002

DOCUMENT-IDENTIFIER: US 20020035621 A1
TITLE: XML-based language description for controlled devices

Summary of Invention Paragraph (4):
[0003] The cost of computing and networking technologies have fallen to the point where computing and networking capabilities can be built into the design of many electronic devices in the home, the office and public places. The combination of inexpensive and reliable shared networking media with a new class of small computing devices has created an opportunity for new functionality based mainly on the connectivity among these devices. This connectivity can be used to remotely control devices, to move digital data in the form of audio, video and still images between devices, to share information among devices and with the unconstrained World Wide Web of the Internet (hereafter "Web") and to exchange structured and secure digital data to support things like electronic commerce. The connectivity also enables many new applications for computing devices, such as proximity-based usage scenarios where devices interact based at least in part on geographical or other notions of proximity. A prevalent feature of these connectivity scenarios is to provide remote access and control of connected devices and services from another device with user interface capabilities (e.g., a universal remote controller, handheld computer or digital assistant, cell phones, and the like). These developments are occurring at the same time as more people are becoming connected to the Internet and as connectivity solutions are falling in price and increasing in speed. These trends are leading towards a world of ubiquitous and pervasive networked computing, where all types of devices are able to effortlessly and seamlessly interconnect and interact.

Detail Description Paragraph (4):
[0046] Universal Plug and Play (UPnP) is an open network architecture that is designed to enable simple, ad hoc communication among distributed devices and services from many vendors. UPnP leverages Internet technology and can be thought of as an extension of the Web model of mobile Web browsers talking to fixed Web servers to the world of peer-to-peer connectivity among mobile and fixed devices. UPnP embraces the zero configuration mantra of Plug and Play (PnP) but is not a simple extension of the PnP host/peripheral model.

Detail Description Paragraph (8):
[0050] The Internet has created a widespread awareness of the value of simple, universal communication that is independent of the underlying transmission technology and independent of technology from any single vendor.

Detail Description Paragraph (10):
[0052] UPnP reuses existing protocols and technology whenever possible. The transition to this highly connected (and connectable) world will not occur overnight. UPnP builds on existing Internet protocols, but accommodates devices that cannot run the complete UPnP protocol suite. UPnP provides an architecture that enables legacy devices to communicate with UPnP devices.

Detail Description Paragraph (11):
[0053] IP internetworking has been chosen as a UPnP baseline due to its proven ability to span different physical media, to enable real world multiple vendor interoperation and to achieve synergy with the Internet and home and office intranets. Internet synergy enables applications such as IP telephony, multiple player games, remote control of home automation and security, Internet based electronic commerce, in addition to simple email and Web browsing. UPnP's scope includes remote control of devices and bulk data transfer, and can be easily extended to specify A/V streaming.

Detail Description Paragraph (25):
[0067] Service Provider. A module used by a UPnP Bridge that translates between UPnP

protocols and the protocols used by Bridged and Legacy Devices. No Service Providers are required for communication among native UPnP devices.

Detail Description Paragraph (49):
[0091] Generic Event Notification Architecture (GENA). An event transport protocol. GENA leverages TCP/HTTP as a transport. GENA has been submitted as an Internet Draft to the IETF. See, J. Cohen, S. Aggarwal, Y. Goland, "General Event Notification Architecture Base: Client to Arbiter", which can be found at http://www.ietf.org/internet-drafts/draft-cohen-gena-clie- nt-00.txt.

Detail Description Paragraph (50):
[0092] Simple Service Discovery Protocol (SSDP). A simple network device discovery protocol. UPnP uses SSDP to allow User Control Points to find Controlled Devices and Services. SSDP operates in a default, completely automatic multicast UDP/IP based mode in addition to a server-based mode that uses TCP/IP for registrations and query. Transitions between the default dynamic mode and server-based mode are automatic and transparent to upper level software. SSDP enables every Controlled Device to control the lifetime that its Description URL is cached in all User Control Points. This enables a Controlled Device to remain visible to User Control Points for a relatively long time (through power cycles), in addition to enabling a Controlled Device to appear and disappear very quickly, all under the control of the Controlled Device. SSDP and related Multicast and Unicast UDP HTTP Messages specifications have been submitted as Internet Drafts to the IETF. See, Y. Goland, "Multicast and Unicast UDP HTTP Messages", which can be found at http://www.ietf.org/int-ernet-drafts/draft-goland-http-udp-00.txt; and Y. Goland, T. Cai, P. Leach., Y. Gu, S. Albright, "Simple Service Discovery Protocol/1.0", which can be found at http://www.ietf.org/internet-drafts/draft-cai-ssdp-- v1-02.txt.

Detail Description Paragraph (80):
[0122] SSDP specifies a default, completely automatic, best-effort multicast UDP-based operating mode, in addition to a server mode that uses TCP for registration and query. Fall-forward to server mode and fallback to the default dynamic mode can occur automatically and transparently as a server is added or removed from a network. Server mode can be used to reduce network traffic, to implement searches based on location or policy and to integrate with a directory system.

Detail Description Paragraph (176):
[0218] Accordingly, the Rehydrator operates as a universal proxy object with data-driven conversion of programmatic interfaces to network data messages. Further, the Rehydrator produces the programmatic interface at the User Control Point based solely on an XML data description. This operation allows the Rehydrator to produce just-in-time transient interfaces to remote device Services without the complexity of code downloads and installation or configuration. Upon a later release of the interface by the application, the Rehydrator destroys the interface without need to de-install or clean up persistent configuration data in a registry or configuration file of the operating system or object execution run-time.

Detail Description Paragraph (243):
[0285] Applications written in C (C Application 604) will be able to utilize the SSDP C API 610 to receive callbacks when notifications are processed by the SSDP service. This is analogous to SSDP clients registering for notifications that services have become available. When a UCP registers for a notification, it passes as a parameter the URL of the service for which it is interested in receiving notifications. This URL is obtained from the description document for that service. (When a service is registered on a UPnP device, it uses this same URL to listen for subscription requests).

Detail Description Paragraph (246):
[0288] The UPnP API 410 is a consumer of the basic C interface provided by the SSDP C API 610 component. In order to integrate seamlessly, the registration of notifications is handled by the Service Object 612 inside the UPnP Object Model. Service objects will register for notifications when they are created. This ensures that the DST is maintained by the UPnP API and is kept up to date. They will implement the callback function required by the registration function. If this callback function is invoked, it will pass on that notification to UCPs. The UCPs can be written in C, C++, VB, or script code, so the mechanism for passing on notifications can be different.

Detail Description Paragraph (276):
[0318] The RegisterUpnpEventSource( ) API gives a GENA client the ability to register itself as an event source. The prototype is as follows:

Detail Description Paragraph (295):
[0337] In UPnP, every UPnP service 210-211 that supports property change event notifications is to be a GENA client. Therefore, when the service is initialized, it must register itself as a GENA event source. It will do this with the RegisterUpnpEventSource( ) API. This returns a handle which can be used in subsequent APIs.

Detail Description Paragraph (353):
[0395] ii. It listens for incoming connection requests on that socket and sets itself up to accept any incoming connections.

Detail Description Paragraph (355):
[0397] i. Calls InternetOpen( ) to get a handle to the internet session

Detail Description Paragraph (373):
[0415] 2. If this is the first time it is called for SSDP_PROPCHANGE notifications, RegisterNotification( ) will call InternetOpen( ) to get a handle to an internet session. This handle is shared among all local UPnP UCPs.

Detail Description Paragraph (397):
[0439] 1. InternetOpen( ) is called if an existing internet session handle does not exist.

Detail Description Paragraph (478):
[0520] The computer 820 operates in a networked environment using logical connections to one or more remote computers, such as a remote computer 849. The remote computer 849 may be a server, a router, a peer device or other common network node, and typically includes many or all of the elements described relative to the computer 820, although only a memory storage device 850 has been illustrated in FIG. 25. The logical connections depicted in FIG. 25 include a local area network (LAN) 851 and a wide area network (WAN) 852. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

Detail Description Paragraph (479):
[0521] When used in a LAN networking environment, the computer 820 is connected to the local network 851 through a network interface or adapter 853. When used in a WAN networking environment, the computer 820 typically includes a modem 854 or other means for establishing communications (e.g., via the LAN 851 and a gateway or proxy server 855) over the wide area network 852, such as the Internet. The modem 854, which may be internal or external, is connected to the system bus 823 via the serial port interface 846. In a networked environment, program modules depicted relative to the computer 820, or portions thereof, may be stored in the remote memory storage device. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

Detail Description Paragraph (485):
[0527] With reference now to FIG. 27, the embedded computing device 100 (FIG. 26) has a software architecture 120 that conforms to the above described UPnP device control model. UPnP provides a mechanism for the embedded computing device to operate in the Internet, as well as networks that have no administrator and no connection to the Internet, and hence no access to configuration services like the Dynamic Host Configuration Protocol (DHCP). DHCP is a mechanism for providing devices with configuration information needed to access the Internet. The mechanism functions through the use of a multicast request for configuration information that is generally responded to with an IP address and DNS server location. Additional information can only be returned in the response.

Detail Description Paragraph (487):
[0529] AutoIP is an enhancement to DHCP that allows devices to claim IP addresses in the absence of a DHCP server or similar IP configuration authority. IP addresses are claimed from a reserved range that is not allowed to be transmitted on the open Internet; thus they are only good for the local network. The embedded computing device 100 claims an address by randomly generating an address in the reserved range and then making an ARP request to see if anyone else has already claimed that address. AutoIP systems will continually check for the presence of a DHCP server so that if one should ever come online, all the AutoIP devices will attempt to switch their IP addresses to one provided by the DHCP server. This allows a network to operate in isolation, be connected to the Internet with DHCP support and then to be returned to isolation. This

type of scenario will be common in homes that use dial-up access.

Detail Description Paragraph (488):
[0530] UPnP also uses the Internet Domain Name System (DNS) for addressing the embedded computing device 900. The DNS is a mapping system that translates human readable domain names, like microsoft.com, into their equivalent IP address. Most corporate intranets implement an internal version of the same technology to provide the same services. In small networks, such as at home or in small business, DNS servers may not exist. Multicast DNS allows DNS requests to be multicast. This allows a machine to see requests for its own name and respond to them. Like AutoIP, Multicast DNS is only used when a DNS server is not available. (For more information, see B. Woodcock, Zocolo, and B. Manning, "Multicast Discovery of DNS Services," which can be found at http://search.ietf.org/internet-drafts/draft-manning-multicast-dns-01.txt- .)

Detail Description Paragraph (490):
[0532] UPnP also provides a Directories mechanism to allow discovery to scale--to the entire Internet if needed. When present, a directory will read all incoming service requests and respond to them itself. This requires that all services (e.g., the embedded computing device 900) register with the directory so that the directory is able to properly answer on their behalf. The directory is also responsible for communicating with other directories in order to determine whether the service is available within the local network, the WAN and potentially the Internet.

Detail Description Paragraph (491):
[0533] To simplify the discovery protocol, directories are treated as proxies. A proxy is a service that accepts requests and takes responsibility for finding the proper response. When a client comes online, it will perform discovery for the proxy. If the proxy is present, then the client will send all future discovery requests to the proxy. If the proxy isn't present, then the client will send all discovery requests to the reserved discovery multicast channel. Regardless of the presence of a proxy, the client's request format and procedures will always be the same. The only difference will be the address to which the client sends its requests. For services, the difference between a proxied and unproxied network is their need to answer discovery requests. On a proxied network, services need do nothing once they have registered with the proxy. On an unproxied network, they answer discovery requests directly.

Detail Description Paragraph (492):
[0534] SSDP uses the UDP- and Transmission Control Protocol (TCP)-based Hyptertext Transport Protocol (HTTP) to provide for service discovery. SSDP uses a Uniform Resource Identifier (URI) to represent the service and the OPTIONS method to provide for discovery. SSDP also will provide support for proxies. These proxies, which are really just fronts for directories, redirect discovery requests to themselves. It is the proxy's job to collect announce requests in order to determine what services are available as well as to communicate with other proxies in order to provide for scalable service discovery.

Detail Description Paragraph (495):
[0537] Accordingly, UPnP supports automatic network introduction, meaning that devices and their related services have the ability to be self-describing and allow automatic configuration. When a device is plugged into the computer network, the device automatically configures itself and acquires a TCP/IP address. The device then announces its presence to other devices already on the network using a simple discovery protocol based on the Internet HTTP protocol and is immediately ready to share its services with any device that requests them.

Detail Description Paragraph (497):
[0539] UPnP Devices support automatic discovery, identification, and configuration to achieve interoperability in the home environment, but must also operate correctly in a managed corporate network. Devices can be networked instead of being attached directly to a PC, and devices are all autonomous citizens on the network, able to talk with each other and exchange information. UPnP provides a unified way of performing directory services with automatic configuration. Capability for simple discovery mechanism used in the home environment provides the ability for any device to become a node on the global Internet. Additionally, directory services can be leveraged if they are available in the corporate environment.

Detail Description Paragraph (498):
[0540] UPnP provides a common set of interfaces for accessing devices and services, enabling the operational unification of diverse media types. Communications protocols

for Universal Plug and Play are based on industry standards, especially key Internet standards such as TCP/IP, HTML, XML, HTTP, DNS, LDAP, and others. Individual implementations for particular networks and buses are built on established protocols.

Detail Description Paragraph (499):
[0541] As shown in FIG. 27, the software architecture 920 of the embedded computing device 900 (FIG. 26) includes the following software code modules that implement UPnP: device functions 922, simple discovery 924, Hypertext Transport Protocol (HTTP) 925, Transmission Control Protocol/Internet Protocol (TCP/IP) stack 926, Dynamic Host Configuration Protocol (DHCP) with AutoIP extension 928, Domain Name System (DNS) with Multicast DNS extension 930, and physical media 910 (also shown in FIG. 26). The device functions 922 is a software code module to implement the device's functionality. For example, where the embedded computing device is a VCR, the device functions code can include code to implement start, stop, pause, record and other functions that the VCR can perform.

Detail Description Paragraph (504):
[0546] The HTTP 925 is a software code modules (about 20 Kbytes) that implements the standard HTTP protocol, which is an open standard mechanism for client/server message-based communication. HTTP provides for proxying, content negotiation and security. (For more information, see R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, "Hypertext Transfer Protocol-HTTP/1.1", which can be found at http://www.ietf.org/rfc/rfc2068.txt.)

Detail Description Paragraph (505):
[0547] The TCP/IP stack 926 implements the standard TCP/IP networking protocols for communication on the computer network. The Internet Protocol (IP) is the foundation protocol of the Internet. It defines how a single message is sent from a source through zero or more routers to its final destination. It covers issues such as message length, message fragmentation, addressing, and routing concerns. The Transmission Control Protocol (TCP) is an IP-based protocol that provides support for the reliable, ordered delivery of messages over IP. Additionally, User Datagram Protocol (UDP) and Internet Group Management Protocol (IGMP) multicast send/listen capability are included in the implementation.

Detail Description Paragraph (506):
[0548] The AutoIP 928 is a software code module also used for automatic network introduction via AutoIP in the UPnP protocol. AutoIP uses a predefined set of IP addresses and, when a device is connected to the network, it pings an address in this address space. If it gets no replies, the device assumes that the address is available and assigns it to itself. To make this functionality even more useful it is combined with Multicast DNS, in which the device itself holds its own name. Thus it is not even necessary to determine what IP address the device assigned to itself, because its name can always be used instead. An IP Multicast is a mechanism for sending a single message to multiple recipients. IP multicasting is especially useful for discovery operations where one does not know exactly who has the information one seeks. In such cases, one can send a request to a reserved IP multicast address. Any services that can provide the requested information will also subscribe to the multicast request and thus be able to hear the information request and properly respond. Multicast DNS is a proposal to the IETF on rules for making normal DNS requests using multicast UDP. (For more information, see B. Woodcock, B. Manning, "Multi cast Discovery of DNS Services", which can be found at http://www.ietf.org/internet-drafts/draft-manning-multicast-dns-01.txt.)

Detail Description Paragraph (507):
[0549] The DHCP 930 is a software code module that implements the Dynamic Host Configuration Protocol (DHCP), which is a mechanism for providing devices with configuration information needed to access the Internet. The mechanism functions through the use of a multicast request for configuration information that is generally responded to with an IP address and DNS server location. Additional information can only be returned in the response.

Detail Description Paragraph (515):
[0557] With reference now to FIG. 30, a client that accesses and uses the embedded computing device 900 over the computer network has an exemplary client software architecture 950, which includes software code modules for applications 952, simple discovery 954, XML 955, LDAP 956, TCP/IP stack 958 and a network interface card (NIC) 960 that provides a physical connection to the computer network. The applications 952 is a software code module that provides a user interface features for locating desired

devices (e.g., embedded computing device 900) and services on the computer network, and also user interface features to interact with the located device or service. The applications 952 can include an Internet browser, such as the Microsoft Internet Explorer, that can present the XML device description in accordance with an associated XSL style sheet for interaction with the embedded computing device and activation of its operational functionality.

Detail Description Paragraph (518):
[0560] FIG. 31 illustrates a pervasive computing environment 1000, such as may be installed in a home, office or public place, which includes a large number of embedded computing devices, such as the illustrated device 900 (FIG. 26). The pervasive computing environment 1000 includes personal computers 1002, 1004 (e.g., of the type shown in FIG. 25) connected via a local area network (LAN) 1006. The PC 1002 is connected via a universal serial bus 1016 to a telephone modem 1010, XDSL interface 1011 or a cable modem 1012, which in turn provide a connection with the computer network, e.g., the Internet.

Detail Description Table CWU (1):
1 User Control Point Controlled Device Function Module Function Module Initiate discovery Discovery Client Respond to Discovery Server of Controlled discovery Devices, requests. Retrieve Description Client Provide Description Description Description Server Documents. Documents. Display a folder Visual Navigation of icons per discovered Device and allow transfer of control to a selected device. View user Web Browser Provide user Presentation interface exposed inteface for (Web) Server by a Controlled remote User Device. Control Points. Execute Application applications. Execution Environment Invoke Rehydrator Accept Control Server incoming Commands on a Commands in plus native Controlled Device SCPs and control logic by sending execute them. Service Control Protocols in response to local API calls. Inform a Event Accept requests Event Controlled Device Subscription for Events and Subscription of a desire to Client remember Server receive Events. them. Receive an Event. Event Sink Send an Event. Event Source

L10: Entry 35 of 39 File: USPT Sep 11, 2001

DOCUMENT-IDENTIFIER: US 6289382 B1
TITLE: System, method and article of manufacture for a globally addressable interface in a communication services patterns environment

Brief Summary Text (2):
The present invention relates to software patterns and more particularly to aiding a system in need of service by locating a service provider capable of delivering the required service, wherein this is accomplished by way of a globally addressable interface.

Brief Summary Text (4):
An important use of computers is the transfer of information over a network. Currently, the largest computer network in existence is the Internet. The Internet is a worldwide interconnection of computer networks that communicate using a common protocol. Millions of computers, from low end personal computers to high-end super computers are coupled to the Internet.

Brief Summary Text (5):
The Internet grew out of work funded in the 1960s by the U.S. Defense Department's Advanced Research Projects Agency. For a long time, Internet was used by researchers in universities and national laboratories to share information. As the existence of the Internet became more widely known, many users outside of the academic/research community (e.g., employees of large corporations) started to use Internet to carry electronic mail.

Brief Summary Text (6):
In 1989, a new type of information system known as the World-Wide-Web ("the Web") was introduced to the Internet. Early development of the Web took place at CERN, the European Particle Physics Laboratory. The Web is a wide-area hypermedia information retrieval system aimed to give wide access to a large universe of documents. At that time, the Web was known to and used by the academic/research community only. There was no easily available tool which allows a technically untrained person to access the Web.

Brief Summary Text (7):
In 1993, researchers at the National Center for Supercomputing Applications (NCSA) released a Web browser called "Mosaic" that implemented a graphical user interface (GUI). Mosaic's graphical user interface was simple to learn yet powerful. The Mosaic browser allows a user to retrieve documents from the World-Wide-Web using simple point-and-click commands. Because the user does not have to be technically trained and the browser is pleasant to use, it has the potential of opening up the Internet to the masses.

Brief Summary Text (8):
The architecture of the Web follows a conventional client-server model. The terms "client" and "server" are used to refer to a computer's general role as a requester of data (the client) or provider of data (the server). Under the Web environment, Web browsers reside in clients and Web documents reside in servers. Web clients and Web servers communicate using a protocol called "HyperText Transfer Protocol" (HTTP). A browser opens a connection to a server and initiates a request for a document. The server delivers the requested document, typically in the form of a text document coded in a standard Hypertext Markup Language (HTML) format, and when the connection is closed in the above interaction, the server serves a passive role, i.e., it accepts commands from the client and cannot request the client to perform any action.

Drawing Description Text (90):
FIG. 88 illustrates the method in which the present invention registers and then

locates a Globally Addressable Interface;

Drawing Description Text (101):
FIG. 99 illustrates a message trace diagram showing the interactions between a Client and a Server using Paging Communication to satisfy the previously mentioned scenario;

Drawing Description Text (104):
FIG. 102 illustrates how a pool can be created that reuses GAI proxies;

Drawing Description Text (105):
FIG. 103 illustrates the implementation of a Refreshable Proxy Pool;

Drawing Description Text (189):
FIG. 187 illustrates a manner in which the present invention registers requests;

Drawing Description Text (191):
FIG. 189 illustrates an Ad Hoc Registration feature;

Detailed Description Text (39):
Thus, through the development of frameworks for solutions to various problems and programming tasks, significant reductions in the design and development effort for software can be achieved. A preferred embodiment of the invention utilizes HyperText Markup Language (HTML) to implement documents on the Internet together with a general-purpose secure communication protocol for a transport medium between the client and the Newco. HTTP or other protocols could be readily substituted for HTML without undue experimentation. Information on these products is available in T. Berners-Lee, D. Connoly, "RFC 1866: Hypertext Markup Language--2.0" (November 1995); and R. Fielding, H, Frystyk, T. Berners-Lee, J. Gettys and J. C. Mogul, "Hypertext Transfer Protocol--HTTP/1.1: HTTP Working Group Internet Draft" (May 2, 1996). HTML is a simple data format used to create hypertext documents that are portable from one platform to another. HTML documents are SGML documents with generic semantics that are appropriate for representing information from a wide range of domains. HTML has been in use by the World-Wide Web global information initiative since 1990. HTML is an application of ISO Standard 8879; 1986 Information Processing Text and Office Systems; Standard Generalized Markup Language (SGML).

Detailed Description Text (51):
Sun's Java language has emerged as an industry-recognized language for "programming the Internet." Sun defines Java as: "a simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, high-performance, multithreaded, dynamic, buzzword-compliant, general-purpose programming language. Java supports programming for the Internet in the form of platform-independent Java applets." Java applets are small, specialized applications that comply with Sun's Java Application Programming Interface (API) allowing developers to add "interactive content" to Web documents (e.g., simple animations, page adornments, basic games, etc.). Applets execute within a Java-compatible browser (e.g., Netscape Navigator) by copying code from the server to client. From a language standpoint, Java's core feature set is based on C++. Sun's Java literature states that Java is basically, "C++ with extensions from Objective C for more dynamic method resolution."

Detailed Description Text (52):
Another technology that provides similar function to JAVA is provided by Microsoft and ActiveX Technologies, to give developers and Web designers wherewithal to build dynamic content for the Internet and personal computers. ActiveX includes tools for developing animation, 3-D virtual reality, video and other multimedia content. The tools use Internet standards, work on multiple platforms, and are being supported by over 100 companies. The group's building blocks are called ActiveX Controls, small, fast components that enable developers to embed parts of software in hypertext markup language (HTML) pages. ActiveX Controls work with a variety of programming languages including Microsoft Visual C++, Borland Delphi, Microsoft Visual Basic programming system and, in the future, Microsoft's development tool for Java, code named "Jakarta." ActiveX Technologies also includes ActiveX Server Framework, allowing developers to create server applications. One of ordinary skill in the art readily recognizes that ActiveX could be substituted for JAVA without undue experimentation to practice the invention.

Detailed Description Text (253):
When deciding whether to employ a Netcentric solution, i.e. incorporating Web-based user interfaces and Internet application styles, keep in mind that these technologies

are not a panacea and should be used only when there is solid business reason. They require new investments in skills, tools, development and operations processes. Due to the relative immaturity of tools and products, they also represent additional risks both in technical terms, such as performance and reliability, and in strategic terms, such as vendor and product quality and stability.

Detailed Description Text (269):
The momentum of the Internet is putting a lot of pressure on vendors of various devices to be web-enabled. Having the Internet infrastructure in place makes it more feasible for vendors to create new physical devices from which electronic information can be accessed. For example, Web televisions are gaining momentum. Now users can access the Internet from a television set. Network Computers, thin-client devices that download and run applications from a centrally maintained server are generating a lot of interest. Also, users want to have access to the same information from multiple physical devices. For example, a user might want to have access to his/her e-mail from a cellular phone, from a Web TV or their portable PC.

Detailed Description Text (367):
Netcentric Computing also called Netcentric Architecture, Netcentric Technology, etc. is an emerging architecture style which expands the reach of computing both within and outside the enterprise. Netcentric enables sharing of data and content between individuals and applications. These applications provide capabilities to publish, interact or transact. Netcentric represents an evolution of Client/Server which may utilize internet technologies to connect employees, customers, and business partners.

Detailed Description Text (448):
Today most Netcentric applications are Web based and are launched from the Web browser. Additionally, one is now beginning to see other types of Netcentric solutions. For example, PointCast is a Netcentric application located on the users machine; it relies on the Internet to deliver stock prices, news headings, sports updates, etc. to the user. However, it is not launched from the Web browser--it is its own application. In the future there will be more Netcentric applications that use this approach for delivering information.

Detailed Description Text (478):
JetForms JetForm Design--provides tools to design, fill, route, print and manage electronic forms, helping organizations reduce costs and increase efficiency by automating processing of forms across local and wide area networks as well as the Internet. Lotus Forms--Lotus Development Corporations electronic forms software provides tools to design, route and track forms to automate business processes for the workgroup or the extended enterprise. Lotus Forms is designed to run with Lotus Notes or as a standalone application. It is comprised of two parts: Forms Designer, an application-development version, and Forms Filler, a runtime version for users. Visual Basic--a development tool that provides a comprehensive development environment for building complex applications.

Detailed Description Text (492):
Is there a need for object registration/de-registration?

Detailed Description Text (496):
Parlez-vous Internet?

Detailed Description Text (498):
Language philosopher Benjamin Whorf once said, "We dissect nature along lines laid down by our native language. Language is not simply a reporting device for experience, but a defining framework for it." This notion is especially true when applied to the World Wide Web. The evolution of the Web from a rigid, text-centric village to an elastic, multimedia-rich universe has been driven by modifications to the languages behind it. The Internet is at a crucial point in its development as a number of enhancements for extending Web technology come under scrutiny by Internet standards groups. These enhancements will ultimately push the Web into the realms of distributed document processing and interactive multimedia.

Detailed Description Text (500):
Although the World Wide Web was not created until the early 1990s, the language behind it dates back to the genesis of the Internet in the 1960s. Scientists at IBM were working on a Generalized Markup Language (GML) for describing, formatting, and sharing electronic documents. Markup refers to the practice in traditional publishing of annotating manuscripts with layout instructions for the typesetters.

Detailed Description Text (510):
Microsoft's Internet Explorer 4.0 supports a W3C "Working Draft" DOM specification that uses the CSS standard for layout control and Web document object manipulation. In contrast, Netscape's implementation of DHTML in Communicator 4.0 uses a proprietary "Dynamic Layers" tag, which assigns multiple layers to a page within which objects are manipulated. As a result, Web pages authored using either version of DHTML may not be viewed properly using the other's browser. XML: X marks the spot

Detailed Description Text (511):
HTML 4.0 and Dynamic HTML have given Web authors more control over the ways in which a Web page is displayed. But they have done little to address a growing problem in the developer community: how to access and manage data in Web documents so as to gain more control over document structure. To this end, leading Internet developers devised Extensible Markup Language (XML), a watered-down version of SGML that reduces its complexity while maintaining its flexibility. Like SGML, XML is a meta-language that allows authors to create their own customized tags to identify different types of data on their Web pages. In addition to improving document structure, these tags will make it possible to more effectively index and search for information in databases and on the Web.

Detailed Description Text (512):
XML documents consist of two parts. The first is the document itself, which contains XML tags for identifying data elements and resembles an HTML document. The second part is a DTD that defines the document structure by explaining what the tags mean and how they should be interpreted. In order to view XML documents, Web browsers and search engines will need special XML processors called "parsers." Currently, Microsoft's Internet Explorer 4.0 contains two XML parsers: a high-performance parser written in C++ and another one written in Java.

Detailed Description Text (513):
A number of vendors plan to use XML as the underlying language for new Web standards and applications. Microsoft uses XML for its Channel Definition Format, a Web-based "push" content delivery system included in Internet Explorer 4.0. Netscape will use XML in its Meta Content Framework to describe and store metadata, or collections of information, in forthcoming versions of Communicator. XML is currently playing an important role the realm of electronic commerce via the Open Financial Exchange, an application developed by Microsoft, Intuit, and CheckFree for conducting electronic financial transactions. Similarly, HL7, a healthcare information systems standards organization, is using XML to support electronic data interchange EDI of clinical, financial, and administrative information (http://www.mcis.duke.edu/standards/HL7/sigs/sgml/index.html).

Detailed Description Text (515):
In 1994, a number of Internet thought leaders, including Tim Berners-Lee--the "father" of the Web--met to determine how they could bring the hot, new technology known as virtual reality VR to the Web. VR refers to the use of computers to create artificial and navigable 3-D worlds where users can create and manipulate virtual objects in real time. This led to the creation of Virtual Reality Modeling Language (VRML--pronounced "ver-mul"). VRML is technically not a markup language because it uses graphical rather than text-based file formats.

Detailed Description Text (517):
VRML is capable of displaying static and animated objects and supports hyperlinks to multimedia formats such as audio clips, video files, and graphical images. As users maneuver through VRML worlds, the landscape shifts to match their movements and give the impression that they are moving through real space. The new VRML 2.0 specification finalized in August 1996 intensifies the immersive experience of VR worlds on the Web by enabling users to interact both with each other and with their surroundings. Other new features supported by VRML 2.0 include richer geometry description, background textures, sound and video, multilingual text, Java applets, and scripting using VBScript and JavaScript. VRML will become a significant technology in creating next-generation Internet application as the language continues to mature and its availability increases.

Detailed Description Text (526):
Exemplary products that may be used to implement this component includes Netscape Navigator; Netscape Communicator; Microsoft Internet Explorer; Netscape LiveWire; Netscape LiveWire Pro; Symantec Visual Cafe; Microsoft Front Page; Microsoft Visual

J++; IBM VisualAge.

Detailed Description Text (529):
Microsoft Internet Explorer (IE)--a Web Browser that is tightly integrated with Windows and supports the major features of the Netscape Navigator as well as Microsoft's own ActiveX technologies.

Detailed Description Text (539):
Plug-in--a term coined by Netscape, a plug-in is a software program that is specifically written to be executed within a browser for the purpose of providing additional functionality that is not natively supported by the browser, such as viewing and playing unique data or media types. Typically, to use a plug-in, a user is required to download and install the Plug-in on his/her client machine. Once the Plug-in is installed it is integrated into the Web browser. The next time a browser opens a Web page that requires that Plug-in to view a specific data format, the browser initiates the execution of the Plug-in. Until recently Plug-ins were only accessible from the Netscape browser. Now, other browsers such as Microsoft's Internet Explorer are beginning to support Plug-in technology as well. Also, Plug-ins written for one browser will generally need to be modified to work with other browsers. Plug-ins are also operating system dependent. Therefore, separate versions of a Plug-in may be required to support Windows, Macintosh, and Unix platforms.

Detailed Description Text (549):
Exemplary products that may be used to implement this component include Real Audio Player; VDOLive; Macromedia Shockwave; Internet Phone; Web 3270.

Detailed Description Text (550):
Real Audio Player--a plug-in designed to play audio and video in real-time on the Internet without requiring to download the entire audio file before you can begin listening, or a video file before you can begin viewing.

Detailed Description Text (552):
Internet Phone--one of several applications which allow two-way voice conversation over the Internet, similar to a telephone call.

Detailed Description Text (553):
Web3270--a plug-in from Information Builders that allows mainframe 3270-based applications to be viewed across the Internet from within a browser. The Web3270 server provides translation services to transform a standard 3270 screen into an HTML-based form. Interest in Web3270 and similar plug-ins has increased with the Internets ability to provide customers and trading partners direct access to an organizations applications and data. Screen scraping plug-ins can bring legacy applications to the Internet or intranet very quickly.

Detailed Description Text (556):
Additionally Microsoft has introduced ActiveX documents which allow Forms such as Word documents, Excel spreadsheets, Visual Basic windows to be viewed directly from Internet Explorer just like HTML pages.

Detailed Description Text (562):
Hyperlink--the Internet has popularized the use of underlined key words, icons and pictures that act as links to further pages. The hyperlink mechanism is not constrained to a menu, but can be used anywhere within a page or document to provide the user with navigation options. It can take a user to another location within the same document or a different document altogether, or even a different server or company for that matter. There are three types of hyperlinks:

Detailed Description Text (566):
Customized Menu--a menu bar with associated pull-down menus or context-sensitive pop-up menus. However, as mentioned earlier this method hides functions and options within menus and is difficult for infrequent users. Therefore, it is rarely used directly in HTML pages, Java applets or ActiveX controls. However, this capability might be more applicable for intranet environments where the browsers themselves need to be customized (e.g., adding custom pull-down menus within Internet Explorer) for the organizations specific business applications.

Detailed Description Text (630):
IBM DB2--the leader in MVS mainframe database management, IBM DB2 family of relational database products are designed to offer open, industrial strength database management

for decision support, transaction processing and line of business applications. The DB2 family now spans not only IBM platforms like personal computers, AS/400 systems, RISC System/6000 hardware and IBM mainframe computers, but also non-IBM machines such as Hewlett-Packard and Sun Microsystems. Microsoft SQL Server--the latest version of a high-performance client/server relational database management system. Building on version 6.0, SQL Server 6.5 introduces key new features such as transparent distributed transactions, simplified administration, OLE-based programming interfaces, improved support for industry standards and Internet integration.

Detailed Description Text (697):
Products such as Lotus Notes and Microsoft Exchange allow remote users to replicate documents between a client machine and a central server, so that the users can work disconnected from the network. When reattached to the network, users perform an update that automatically exchanges information on new, modified and deleted documents.

Detailed Description Text (714):
Microsoft Index Server 1.1--allows for search of Web documents, including Microsoft Word and Microsoft Excel. It works with Windows NT Server 4.0 and Internet Information Server 2.0 or higher to provide access to documents stored on an intranet or Internet site. Index Server supports full-text searches and retrieves all types of information from the Web browser including HTML, text, and all Microsoft Office documents, in their original format.

Detailed Description Text (715):
Netscape Catalog Server 1.0--provides an automated search and discovery server for creating, managing, and keeping current an online catalog of documents residing on corporate intranets and the Internet. Catalog Server offers query by full text, category, or attributes such as title, author, date, etc. It also supports multiple file formats, including HTML, Word, Excel, PowerPoint, and PDF.

Detailed Description Text (723):
Virtual Resource services proxy or mimic the capabilities of specialized, network connected resources. This allows a generic network node to emulate a specialized physical device. In this way, network users can interface with a variety of specialized resources.

Detailed Description Text (744):
Virtual Resource services proxy or mimic the capabilities of specialized, network-connected resources. This allows a generic network node to emulate a specialized physical device. In this way, network users can interface with a variety of specialized resources. An examples of a Virtual Resource service is the capability to print to a network printer as if it were directly attached to a workstation.

Detailed Description Text (774):
Common Internet File System (CIFS)--an enhancement to SMB for distributed file systems in a TCP/IP environment.

Detailed Description Text (804):
Internet Telephony--Internet telephony products enable voice telephone calls (and faxing, voice mail retrieval, etc.) through the Internet. For example, an Internet telephony product can accept voice input into a workstation, translate it into an IP data stream, and route it through the Internet to a destination workstation, where the data is translated back into audio.

Detailed Description Text (807):
Lucent PassageWay; COM2001s TransCOM; NetSpeaks WebPhone; VocalTecs Internet Phone; IDTs Net2Phone; Octel Communications Unified Messenger

Detailed Description Text (811):
The following are examples of Internet telephony products:

Detailed Description Text (813):
VocalTec's Internet Phone

Detailed Description Text (825):
rlogin--a remote terminal service implemented under BSD UNIX. The concept behind rlogin is that it supports "trusted" hosts. This is accomplished by having a set of machines that share common file access rights and logins. The user controls access by authorizing remote login based on a remote host and remote user name.

Detailed Description Text (883):
Lightweight Directory Access Protocol (LDAP) a de facto standard for accessing X.500-compatible directory information in an Internet/intranet environment.

Detailed Description Text (886):
Another protocol that has emerged is the Lightweight Directory Access Protocol (LDAP), which is a slimmed-down version of the X.500 directory client and is seen as a possible replacement for X.500. LDAP is a standard protocol for accessing and updating directory information in a client/server environment; it has evolved into an emerging standard for directory replication for the Internet, and is backed by vendors such as Netscape, Novell, Microsoft, IBM and AT&T that can provide low-level compatibility among directory systems.

Detailed Description Text (897):
Domain Name Service--The most common and widely used Name Service on the Internet is Domain Name Service (DNS) which resolves a pronounceable name into an IP address and vice versa. For instance, DNS could resolve the domain name of www.ac.com to be 204.167.146.195. DNS functionality is distributed across many computers within the network.

Detailed Description Text (898):
Microsoft's Windows Internet Name Service (WINS)--WINS is Microsoft's proprietary method for mapping IP addresses to NetBIOS device names. WINS works with Windows 3.x, Windows 95, and Windows NT clients.

Detailed Description Text (908):
TCP/IP--Transmission Control Protocol/Internet Protocol (TCP/IP) is the principle method for transmitting data over the Internet today. This protocol is responsible for ensuring that a series of data packets sent over a network arrive at the destination and are properly sequenced.

Detailed Description Text (968):
File Transfer Protocol (FTP) allows users to upload and download files across the network. FTP also provides a mechanism to obtain filename, directory name, attributes and file size information. Remote file access protocols, such as Network File System (NFS) also use a block transfer method, but are optimized for online read/write paging of a file.

Detailed Description Text (970):
Secure Hypertext Transfer Protocol (S-HTTP)--a secure form of HTTP, mostly for financial transactions on the Web. S-HTTP has gained a small level of acceptance among merchants selling products on the Internet as a way to conduct financial transactions (using credit card numbers, passing sensitive information) without the risk of unauthorized people intercepting this information. S-HTTP incorporates various cryptographic message formats such as DSA and RSA standards into both the Web client and the Web server.

Detailed Description Text (979):
Computer Associates' CA-XCOM--provides data transport between mainframes, midrange, UNIX, and PC systems. XcelleNet's RemoteWare--retrieves, appends, copies, sends, deletes, and renames files between remote users and enterprise systems. Hewlett-Packard's HP FTAM--provides file transfer, access, and management of files in OSI networks.

Detailed Description Text (1007):
Publish and Subscribe (also known as Push messaging)--as shown in FIG. 19, is a special type of data delivery mechanism that allows processes to register an interest in (i.e., subscribe to) certain messages or events. An application then sends (publishes) a message, which is then forwarded to all processes that subscribe to it.

Detailed Description Text (1022):
A new Internet gateway that allows customers and partners to run mission critical business applications over an unreliable network.

Detailed Description Text (1044):
TIB/Rendezvous' publish/subscribe technology is the foundation of TIBnet, TibCos solution for providing information delivery over intranets, extranets and the Internet. It is built upon The Information Bus.RTM. (TIB.RTM.) software, a highly scaleable

messaging middleware technology based on an event-driven publish/subscribe model for information distribution. Developed and patented by TIBCO, the event-driven, publish/subscribe strategy allows content to be distributed on an event basis as it becomes available. Subscribers receive content according to topics of interest that are specified once by the subscriber, instead of repeated requests for updates. Using IP Multicast, TIBnet does not clog networks, but instead, provides for the most efficient real-time information delivery possible.

Detailed Description Text (1046):
Streaming is the process of transferring time-sensitive data streams (e.g., video and/or audio) in real-time. Streaming differs from the other types of Core Messaging services in that it delivers a continuous, one-way stream of data, rather than the relatively short messages associated with RPC and Message-Oriented Middleware messaging or the large, batch transfers associated with File Transfer. (While the media stream is one-way from the server to the client, the client can issue stream controls to the server.) Streaming may be used to deliver video, audio, and/or other real-time content across the Internet or within enterprise networks.

Detailed Description Text (1048):
Real-time Streaming Protocol (RTSP)--RTSP is a draft Internet protocol for establishing and controlling on-demand delivery of real-time data. For example, clients can use RTSP to request specific media from a media server, to issue commands such as play, record and pause, and to control media delivery speed. Since RTSP simply controls media delivery, it is layered on top of other protocols, such as the following.

Detailed Description Text (1068):
SMTP--Simple Mail Transfer Protocol (SMTP) is a UNIX/Internet standard for transferring e-mail among servers.

Detailed Description Text (1069):
MIME--Multi-Purpose Internet Mail Extensions (MIME) is a protocol that enables Internet users to exchange multimedia e-mail messages.

Detailed Description Text (1071):
IMAP4--Internet Message Access Protocol, Version 4 (IMAP4) allows a client to access and manipulate electronic mail messages on a server. IMAP4 permits manipulation of remote message folders, called "mailboxes", in a way that is functionally equivalent to local mailboxes. IMAP4 also provides the capability for an off-line client to re-synchronize with the server. IMAP4 includes standards for message handling features that allow users to download message header information and then decide which e-mail message contents to download.

Detailed Description Text (1074):
The Multi-part Internet Mail Extensions (MIME) standard has gained acceptance as the Internet mechanism for sending E-mail containing various multimedia parts, such as images, audio files, and movies. S/MIME, or secure MIME adds encryption and enables a secure mechanism for transferring files.

Detailed Description Text (1075):
Although currently POP3 is the popular Internet E-Mail message handling protocol, recently the lesser known IMAP4 protocol has been gaining in adoption among mail server and mail client software providers. IMAP was designed to add features beyond POP that allow users to store and archive messages and support mobile users that need to keep messages on a central server as well as on their laptop.

Detailed Description Text (1076):
Organizations are looking to use vehicles like E-Mail and the Internet to enable communications with customers and trading partners. The least common denominator E-mail capability today is very rudimentary (ASCII text). But as the standards listed here as well as others become integrated into most of the popular E-mail products and gateways this will change enabling a more flexible and useful commercial communications medium.

Detailed Description Text (1079):
The following E-Mail products are based on the open Internet standards defined above:

Detailed Description Text (1080):
Netscape Mail Server--Netscapes implementation of an open standards-based client/server messaging system that lets users exchange information within a company as well as across the Internet. It includes support for all standard protocols, and is packaged

with Netscapes SuiteSpot server line.

Detailed Description Text (1081):
Post.Office--one of the leading POP3/SMTP mail servers for the Internet community as well as corporate intranets. This message transport agent is based entirely on the open standards of the Internet, ensuring maximum compatibility with other systems.

Detailed Description Text (1084):
Lotus Notes--platform-independent client/server mail system. Notes Mail can support over 1,500 active users per server, offering Internet integration, distributed replication and synchronization. Lotus Notes also provides integrated document libraries, workflow, calendaring and scheduling, and a cc:Mail user interface.

Detailed Description Text (1085):
Microsofts Exchange Server--Exchange 4.0 provides a messaging and groupware platform to support collaboration solutions on Windows machines. Microsoft Exchange 5.0 has support for all of the key Internet protocols. These include POP3 for mailbox access, SMTP for mail sending and receiving, NNTP for newsgroups and discussion forums, LDAP for directory access, HTTP and HTML for access via a web browser, and SSL for security.

Detailed Description Text (1092):
UUcoding--process for converting 8-bit binary files into 7-bit ASCII files for transmission via e-mail over the Internet (the Internet only supports seven bit characters in e-mail messages); UUencode and UUdecode utilities on end nodes perform the conversion.

Detailed Description Text (1137):
The OMGs Internet Inter-Orb Protocol (IIOP) specifies a set of message formats and common data representations for communication between ORBs over TCP/IP networks. CORBA-based Object Messaging is summarized in FIG. 21.

Detailed Description Text (1188):
EDI messages have traditionally been sent between companies using a VAN (Value Added Network). VANs have been criticized for their relatively high cost in comparison to public networks like the Internet. Recently, EDI messaging vendors such as Premenos have been creating software with built-in encryption features to enable companies to send EDI transmissions securely over the Internet.

Detailed Description Text (1214):
SafePassage is a full-strength, encrypting Web proxy. It is designed to supplement the security of browsers whose authentication and encryption capabilities have been weakened to comply with United States export regulations. For these types of browsers, SafePassage will provide client authentication certificates and full-strength encryption (128 bit).

Detailed Description Text (1225):
The advantage of SSL over S/HTTP is that SSL is not restricted to HTTP but can also be used for securing other TCP/IP based services such as FTP, Telnet, etc. SSL can provide session level data encryption and authentication to enable secure data communications over public networks such as the Internet.

Detailed Description Text (1226):
The need for Encryption Services is particularly strong where electronic commerce solutions that involve exchanging sensitive or financial data are to be deployed over public networks such as the Internet. Cryptography can be used to achieve secure communications, even when the transmission media (for example, the Internet) is untrustworthy. Encryption Services can also be used to encrypt data to be stored (e.g., sensitive product information on a sales person's laptop) to decrease the chance of information theft.

Detailed Description Text (1241):
Filters--World Wide Web filters can prevent users from accessing specified content or Internet addresses. Products can limit access based on keywords, network addresses, time-of-day, user categories, etc.

Detailed Description Text (1242):
Application Proxies--An application-level proxy, or application-level gateway, is a robust type of firewall. (A firewall is a system that enforces an access control policy between a trusted internal network and an untrusted external network.) The application

proxy acts at the application level, rather than the network level. The proxy acts as a go-between for the end-user by completing the user-requested tasks on its own and then transferring the information to the user. The proxy manages a database of allowed user actions, which it checks prior to performing the request.

Detailed Description Text (1245):
Microsoft Windows NT; Novell Netware; UNIX; Check Points Firewall-1; Raptor Systems Eagle Firewall; Microsoft Proxy Server; Netscape Proxy Server; Microsystem Softwares Cyber Patrol Corporate; Net Nanny Softwares Net Nanny

Detailed Description Text (1248):
Application Proxies

Detailed Description Text (1249):
Microsoft Proxy Server--allows for designation of who can access the Internet and which services they can use. Administrators can establish additional credentials for logging on, set specific dialing hours or days of the week, and restrict access to certain sites altogether.

Detailed Description Text (1250):
Netscape Proxy Server--high-performance server software for replicating and filtering access to Web content on the Internet or an intranet. Provides access control, URL filtering, and virus scanning.

Detailed Description Text (1252):
Check Point FireWall-1--combines Internet, intranet and remote user access control with strong authentication, encryption and network address translation (NAT) services. The product is transparent to network users and supports multiple protocols.

Detailed Description Text (1253):
BorderWare Firewall--protects TCP/IP networks from unwanted external access as well as provides control of internal access to external services; supports packet filters and application-level proxies.

Detailed Description Text (1259):
Authentication for accessing resources across an Internet or intranet is not as simple and is a rapidly evolving area. When building e-commerce Web sites there may be a need to restrict access to areas of information and functionality to known customers or trading partners. More granular authentication is required where sensitive individual customer account information must be protected from other customers.

Detailed Description Text (1262):
ID/Password Encryption--offers a somewhat higher level of security by requiring that the user name and password be encrypted during transit. The user name and password are transmitted as a scrambled message as part of each request because there is no persistent connection open between the Web client and the Web server.

Detailed Description Text (1328):
IP (Internet Protocol)

Detailed Description Text (1348):
Policy Routing Protocols--allow Internet backbone routers to accept routing information from neighboring backbone providers on the basis of contracts or other non-technical criteria; routing algorithms are Distance Vector.

Detailed Description Text (1367):
Lucent's Definity; Nortels Meridian; Lucents E5S; Nortels DMS; Tellabs Titan products; Lucents DSX products; Alcatels SX products; AltiGens AltiServ; Lucents Internet Telephony Server

Detailed Description Text (1381):
Lucent's Internet Telephony Server--server software that routes calls from PBXs over the Internet or intranets.

Detailed Description Text (1412):
Check Point FireWall-1--combines Internet, intranet and remote user access control with strong authentication, encryption and network address translation (NAT) services. The product is transparent to network users and supports multiple protocols.

Detailed Description Text (1413):
Secure Computing's BorderWare Firewall Server protects TCP/IP networks from unwanted external access as well as provides control of internal access to external services; supports packet filters and application-level proxies.

Detailed Description Text (1428):
connection establishment delay--time between the connection request and a confirm being received by the requester

Detailed Description Text (1442):
Resource Reservation Protocol (RSVP)--The emerging RSVP specification, proposed by the Internet Engineering Task Force (IETF), allows applications to reserve router bandwidth for delay-sensitive IP traffic. With RSVP, QoS is negotiated for each application connection. RSVP enables the network to reserve resources from end to end, using Frame Relay techniques on Frame Relay networks, ATM techniques on ATM, and so on. In this way, RSVP can achieve QoS across a variety of network technologies, as long as all intermediate nodes are RSVP-capable.

Detailed Description Text (1581):
Microsofts Transaction Server (Viper)--a component-based transaction processing system for developing, deploying, and managing high performance, and scalable enterprise, Internet, and intranet server applications. Transaction Server defines an application programming model for developing distributed, component-based applications. It also provides a run-time infrastructure for deploying and managing these applications.

Detailed Description Text (1679):
Profile Management Services are used to access and update local or remote system, user, or application profiles. User profiles, for example, can be used to store a variety of information such as the user's language and color preferences to basic job function information which may be used by Integrated Performance Support or Workflow Services.

Detailed Description Text (1719):
The popularity of the Internets HTTP protocol has revived the potential need for implementing some form of Context Management Services (storing state information on the server). The HTTP protocol is a stateless protocol. Every connection is negotiated from scratch, not just at the page level but for every element on the page. The server does not maintain a session connection with the client nor save any information between client exchanges (i.e., web page submits or requests). Each HTTP exchange is a completely independent event. Therefore, information entered into one HTML form must be saved by the associated server application somewhere where it can be accessed by subsequent programs in a conversation.

Detailed Description Text (1752):
1. ActiveX/OLE--ActiveX and Object Linking and Embedding (OLE) are implementations of COM/DCOM. ActiveX is a collection of facilities forming a framework for components to work together and interact. ActiveX divides the world into two kinds of components: controls and containers. Controls are relatively independent components that present well defined interfaces or methods that containers and other components can call. Containers implement the part of the ActiveX protocol that allows for them to host and interact with components--forming a kind of back plane for controls to be plugged into. ActiveX is a scaled-down version of OLE for the Internet. OLE provides a framework to build applications from component modules and defines the way in which applications interact using data transfer, drag-and-drop and scripting. OLE is a set of common services that allow components to collaborate intelligently.

Detailed Description Text (1753):
In creating ActiveX from OLE 2.0, Microsoft enhanced the framework to address some of the special needs of Web style computing. Microsofts Web browser, Internet Explorer, is an ActiveX container. Therefore, any ActiveX control can be downloaded to, and plugged into the browser. This allows for executable components to be interleaved with HTML content and downloaded as needed by the Web browser.

Detailed Description Text (1759):
ONE--Open Network Environment (ONE) is an object-oriented software framework from Netscape Communications for use with Internet clients and servers, which enables the integrating of Web clients and servers with other enterprise resources and data. By supporting CORBA, ONE-enabled systems will be able to link with object software from a wide array of vendors, including IBM, Sun Microsystems, Digital Equipment, and Hewlett-Packard. Netscape is positioning ONE as an alternative to Microsoft's

Distributed Common Object Model (DCOM). ONE also complies with Sun Microsystems Java technology.

Detailed Description Text (1769):
There is an explosion of components available in the market place and the ease of accessing and down loading components from the Internet; the decision to buy or build a component is as real as ever. In general clients expect more justification of a build decision v. a buy decision. Feel out the client and the expectations and requirements they may have.

Detailed Description Text (1770):
Components are a viable option and should be researched, even includingeven simple UI controls available on the Internet. Look at market trends to determine which applications/components can meet the bulk of the system needs.

Detailed Description Text (1777):
FIG. 28 illustrates the Base Services of the Netcentric Architecture Framework. Base Services provide server-based support for delivering applications to a wide variety of users over the Internet, intranet, and extranet. The information about these services in the Netcentric framework may be limited based on the least common denominator. For more detailed information about these components refer also to the following frameworks in SAF and/or DAF.

Detailed Description Text (1781):
Web Server Services enable organizations to manage and publish information and deploy Netcentric applications over the Internet and intranet environments. These services support the following:

Detailed Description Text (1787):
Netscape Enterprise Web Server; Microsoft Internet Information Server (IIS); Oracle WebServer

Detailed Description Text (1790):
An enterprise-strength Web server that enables organizations to manage and publish their information and deploy Netcentric applications. Netscape Enterprise Web Server is built on open Internet standards that enable information and applications to scale easily. Supports S-HTTP, Java, and SNMP.

Detailed Description Text (1791):
Microsoft Internet Information Server (IIS)

Detailed Description Text (1796):
Push/Pull Services allow for interest in a particular piece of information to be registered and then changes or new information to be communicated to the subscriber list. Traditional Internet users "surf" the Web by actively moving from one Web page to another, manually searching for content they want and "pulling" it back to the desktop via a graphical browser. But in the push model, on which subscription servers are based on, content providers can broadcast their information directly to individual users'desktops. The technology uses the Internet's strengths as a two-way conduit by allowing people to specify the type of content they want to receive. Content providers then seek to package the requested information for automatic distribution to the user's PC.

Detailed Description Text (1797):
Depending upon requirements, synchronous or asynchronous push/pull services may be required. Synchronous push/pull services provide a mechanism for applications to be notified in real time if a subscribed item changes (e.g., a stock ticker). Asynchronous push/pull services do not require that a session-like connection be present between the subscriber and the information. Internet ListServers are a simple example. Subscribers use e-mail to register an interest in a topic and are notified via e-mail when changes occur or relevant information is available. Asynchronous push/pull services can be useful for pro-actively updating customers on changes in order status or delivering information on new products or services they have expressed an interest in.

Detailed Description Text (1799):
Castanet from Marimba--distributes and maintains software applications and content within an organization or across the Internet, ensuring subscribers always have the most up-to-date information automatically.

Detailed Description Text (2024):
Currently, Internet applications house the majority of the business processing logic on the server, supporting the thin-client model. However, as technology evolves, this balance is beginning to shift, allowing business logic code bundled into components to be either downloaded at runtime or permanently stored on the client machine. Today, client side business logic is supported through the use of Java applets, JavaBeans, Plug-ins and JavaScript from Sun/Netscape and ActiveX controls and VBScript from Microsoft.

Detailed Description Text (2030):
Currently, Internet applications house the majority of the business processing logic on the server, supporting the thin-client model. However, as technology evolves, this balance is beginning to shift, allowing business logic code bundled into components to be either downloaded at runtime or permanently stored on the client machine. Today, client side business logic is supported through the use of Java applets, JavaBeans, Plug-ins and JavaScript from Sun/Netscape and ActiveX controls and VBScript from Microsoft.

Detailed Description Text (2152):
Physical components play a critical role in net-centric computing because they can be distributed, as encapsulated units of executable software, throughout a heterogeneous environment such as the Internet. They have the ability to make the Web more than a toy for retrieving and downloading information. Robert Orfali, Dan Harkey, and Jeri Edwards, well-known experts in the field of component- and object-based development, wrote the following about distributed objects (same as "distributed components" for the purpose of this discussion):

Detailed Description Text (2153):
The next-generation Web--in its Internet, intranet, and extranet incarnations--must be able to deal with the complex requirements of multi-step business-to-business and consumer-to-business transactions. To do this, the Web must evolve into a full-blown client/server medium that can run your line-of-business applications (i.e., a delivery vehicle for business transaction processing) . . . To move to the next step, the Web needs distributed objects.

Detailed Description Text (2836):
Desktop tools today generally include an office suite, drawing tools, case tools and more recently, a web browser. For example, one might find a tool selection like Microsoft Office for documentation, Visio for custom deliverables, Rational Rose for models, and Internet Explorer for viewing HTML versions of the documentation. VBA has become the glue for extending and connecting the information between these tools. Other strategies have included using Notes as a repository for all of the deliverables that users could access information.

Detailed Description Text (2978):
ISO New England energy eXchange. A net-centric internet system build for managing functions associated with a competitive energy market. The energy eXchange is implemented in Java across client and server components and using CORBA as a communications architecture.

Detailed Description Text (3113):
The cost of locating a remote service and establishing a connection to that service can also be a costly endeavor. The Refreshable Proxy Pool pattern describes a robust and efficient way to minimize this "lookup" activity.

Detailed Description Text (3172):
Once all the operations have been grouped into an interface, the interface is given a name appropriate to the operations it bundles. Then the interfaces are announced using the Naming pattern. The Naming pattern enables registration of interfaces in a globally available naming service. FIG. 74 illustrates a customer server 7400 publicly announcing its interfaces 7402.

Detailed Description Text (3173):
Until that time, a client can't find the operations and can't use them. Thus, the Server must use the Lookup or Naming pattern to register its interfaces (not methods). Once the interfaces have been registered with such a service, any client can go to the Naming Service, locate an interface, and access an operation in that interface.

Detailed Description Text (3179):

The scenario was broken into two message trace diagrams. The first message trace sets the stage for the second. In the first message trace, the Server registers two Globally Addressable Interfaces with a Naming Service. The Client then "looks-up" an interface and establishes a connection to that interface.

Detailed Description Text (3187):
2. The client instantiates a Proxy (Browsing Interface Proxy) to the Browsing Interface on the Customer Server.

Detailed Description Text (3188):
3. The Proxy "looks up" the network location of the Browsing Interface. It makes a request of the Naming Service. It requests the network location of the Browsing Interface.

Detailed Description Text (3189):
4. The Naming Service returns the Remote Object Reference (network location) for the Browsing Interface. The Proxy now has all the information it needs to access an operation on the Browsing Interface.

Detailed Description Text (3193):
5. The Client asks the Browsing Interface Proxy for the data associated with customer 1234.

Detailed Description Text (3194):
6. The Browsing Interface Proxy forwards the request across the network to the Browsing Interface.

Detailed Description Text (3199):
11. The Customer Structure is forwarded through the Browsing Interface, across the network and back to the Browsing Interface Proxy.

Detailed Description Text (3200):
12. The Browsing Interface Proxy forwards the Customer Structure to the Client. The Client can now display the data in a UI for a user.

Detailed Description Text (3207):
The GAI class is actually represented by two different classes (and the Component itself). Each GAI is made up of a Proxy and a Skeleton. The Proxy represents the interface on a Client while the Skeleton represents the interface on a Server.

Detailed Description Text (3209):
Proxy--The proxy pattern is generally used to communicate from a Client to a Globally Addressable Interface on a Server.

Detailed Description Text (3210):
Structure Based Communication--Often times, a client needs to display data in a UI for a user (e.g. Customer Information, Order Information, etc.). When communicating through a Globally Addressable Interface, this data is transmitted from the Server to the Client using Structure Based Communication.

Detailed Description Text (3212):
Proxy Pool--The Proxy Pool pattern helps balance the cost of instantiating Remote Proxies and retaining Proxy "freshness." The Proxy Pool pattern can be used to create a pool of Proxies to Globally Addressable Interfaces.

Detailed Description Text (3215):
Naming--The Naming pattern describes a pattern for registering and finding services or objects etc. where they can be easily found in an application. The Naming pattern is often used to register Globally Addressable Interfaces so they are publicly available

Detailed Description Text (3239):
The Client (8006) is the application running on the user's machine. It is responsible for UI presentation, local business objects, and communication using client resident proxies.

Detailed Description Text (3241):
The Components (8010) in this FIG. 80 represents the server components. These are the business entity components and the business process components. They are invoked from the Client via client proxies.

Detailed Description Text (3272):
Proxy--This pattern is documented in Design Patterns by Gamma, Helm, Johnson and Vlissides. The proxy pattern is often used to communicate with server components in a distributed environment. The Proxy would be used to communicate across the Component Integration Architecture to a Legacy Wrapper.

Detailed Description Text (3277):
In an option, the use of the locally addressable interfaces may be facilitated by structured-based communication. As another option, the access may be allowed via a customer interface proxy, a customer server and a database of the globally addressable interface.

Detailed Description Text (3278):
In one embodiment, a request may be received by the customer interface proxy for a reference to one of the locally addressable interfaces. The request may then be forwarded across a network to the database of a server of the globally addressable interface. Also, data from the database may be returned in response to the request. Additionally, an object may be instantiated and populated it with the data by the server of the globally addressable interface. The object may also be associated with one of the locally addressable interfaces. Also, the locally addressable interface may be forwarded to the globally addressable interface. As even a further option, a reference may be forwarded to the locally addressable interface across the network and to the customer interface proxy. In addition, the use of the customer interface proxy may be also used to access the locally addressable interface across the network.

Detailed Description Text (3302):
The scenario was broken into two message trace diagrams. The first message trace sets the stage for the second. In the first message trace, the Server registers a Globally Addressable Interface with the Naming Service. The Globally Addressable Interface will be used to get the Locally Addressable Interface.

Detailed Description Text (3306):
FIG. 88 illustrates the method in which the present invention registers and then locates a Globally Addressable Interface 8800. The various steps shown in FIG. 88 are set forth hereinbelow.

Detailed Description Text (3309):
2. The client instantiates a Proxy (Customer Interface Proxy) to the Customer Interface on the Customer Server.

Detailed Description Text (3310):
3. The Proxy "looks up" the network location of the Customer Interface. It makes a request of the Naming Service. It requests the network location of the Customer Interface.

Detailed Description Text (3311):
4. The Naming Service returns the Remote Object Reference (network location) for the Customer Interface. The Proxy now has all the information it needs to access an operation on the Customer Interface.

Detailed Description Text (3315):
5. The Client asks the Customer Interface Proxy for a reference to a Locally Addressable Interface for Cusomer 1234.

Detailed Description Text (3316):
6. The Customer Interface Proxy forwards the request across the network to the Customer Interface.

Detailed Description Text (3321):
11. The Customer Interface forwards a reference to the Locally Addressable Interface, across the network and back to the Customer Interface Proxy.

Detailed Description Text (3322):
12. The Customer Interface Proxy instantiates an Update Interface Proxy with the reference to the Update Interface.

Detailed Description Text (3323):
13. The Customer Interface Proxy forwards the Update Interface Proxy to the Client.

Detailed Description Text (3324):
14. The Client sends a new address for the customer to the Update Interface Proxy.

Detailed Description Text (3325):
15. The Update Interface Proxy forwards the information across the network to the Update Interface.

Detailed Description Text (3328):
Proxy--The proxy pattern is generally used to communicate from a Client to a Locally Addressable Interface on a Server.

Detailed Description Text (3331):
Structured Based Communications--Often times, a client needs to display data in a UI for a user (e.g. Customer Information, Order Information, etc.). When communicating through a Locally Addressable Interface, this data is transmitted from the Server to the Client using Structure Based Communication.

Detailed Description Text (3353):
Proxy. A Proxy is a placeholder that can accept requests meant for another object. This is typically used in distributed systems when one component wants to send a request to another. Thus, a proxy is often used to make requests of servers that may return null structures.

Detailed Description Text (3381):
FIG. 99 illustrates a message trace diagram showing the interactions between a Client 9900 and a Server 9902 using Paging Communication to satisfy the previously mentioned scenario.

Detailed Description Text (3427):
Context isn't generally stored on the Server when implementing Paging Communication. As a result, it is important to request a minimum collection of data from the server. Most of the relational database are using a count mechanism that defines the maximum number of data to search. That will minimize CPU and memory usage.

Detailed Description Text (3430):
Proxy--The Proxy pattern is often used to communicate between Clients and Servers in a distributed environment. A Proxy is often used to make requests for a "page" of data from a Server.

Detailed Description Text (3435):
Refreshable Proxy Pool

Detailed Description Text (3436):
FIG. 100 illustrates a flowchart for a method 10000 for interfacing a naming service and a client with the naming service allowing access to a plurality of different sets of services from a plurality of globally addressable interfaces. In operation 10002, the naming service calls for receiving locations of the global addressable interfaces. As a result of the calls, proxies are generated based on the received locations of the global addressable interfaces in operation 10004. The proxies are received in an allocation queue where the proxies are then allocated in a proxy pool (see operations 10006 and 10008). Access to the proxies in the proxy pool is allowed for identifying the location of one of the global addressable interfaces in response to a request received from the client in operation 100010.

Detailed Description Text (3437):
The proxy pool may employ load balancing. As another option, the proxies in the proxy pool may be renewed based on an age thereof As a third option, a handle may interface the proxy pool and the client. This handle may additionally interface a plurality of the proxy pools and the client.

Detailed Description Text (3440):
A GAI retrieved by a client will usually go through three phases: 1) Initial retrieval of a GAI from the Trader Service that is subsequently wrapped up in a proxy, 2) Invocations of businesses functions supported by the GAI and 3) Release of the GAI proxy. This often means a long-lived client will repeatedly ask the Trader Service for the same type of interface during its lifetime.

Detailed Description Text (3443):

Therefore, use a Refreshable Proxy Pool mechanism that standardizes the usage, allocation and replenishment of proxies in a client's pool. Initially, the Proxy Pool will allocate a bunch of Proxies to remote services using some sort of a Lookup Service (e.g. Trader Service, Naming Service). The Proxy Pool will hold onto these Proxies and allocate them to Clients as they need them. When the client asks the Proxy Pool for a proxy, the pool will hand out a new Proxy.

Detailed Description Text (3444):
FIG. 102 illustrates how a pool 102001 can be created that reuses GAI proxies.

Detailed Description Text (3445):
In order to balance the system load evenly, the Proxy Pool should implement a Load Balancing approach (e.g. Round Robin) for handing out proxies within the pool.

Detailed Description Text (3446):
Each proxy in the pool has a "retirement age." The "retirement age" determines the time to refresh a given Proxy. When a Proxy reaches its retirement age, it is taken out of the pool and replaced with a freshly allocated Proxy. This ensures the Proxy Pool is refreshed regularly with new GAIs retrieved from the Trader Service. The retirement mechanism helps dynamically balance the systems load.

Detailed Description Text (3453):
Dynamic. Client threads will not have to worry about allocating retired or bad proxies.

Detailed Description Text (3454):
Robustness. By pooling GAI location information, clients are less susceptible to Trader Service failure. This is because the Proxy Pool can operate using the GAIs it has information on for as long as those references main valid.

Detailed Description Text (3455):
The Proxy Pool should be packaged and distributed to client developers so that it is non-intrusive and easy for them to use.

Detailed Description Text (3457):
The client thread using the proxy should not pay a penalty for pooling or allocation.

Detailed Description Text (3460):
FIG. 103 illustrates the implementation of a Refreshable Proxy Pool 10300. The Refreshable Proxy Pool is based on a pool-queue approach. In this design, the pool holds allocated proxies 10302 while the queue allocates and replenishes the pool with proxies. To handle the allocation and replenishment, a worker or allocation thread 10304 runs on the queue and makes calls to the Trader Services as needed.

Detailed Description Text (3461):
There can be numerous proxy pools, but this implementation supports typed pools Using C++ templates, i.e., each pool will only contain proxies of one type. This allows the client to create a class that is passed to the proxy pool and supports client specific properties in the pool such as pool size, retirement age, etc. Also, due to synchronization issues with the rest of the architecture, there can be only one allocation queue.

Detailed Description Text (3462):
Clients who wish to use a pooled proxy will create a handle as a wrapper. This handle wrapper takes care of the problems associated with sharing resources across threads such as lazy initialization, reference counting, allocation, and de-allocation.

Detailed Description Text (3463):
Handles are classes that abstract the users away from the implementation. Handles are generally stack based and exist for the lifetime of a method invocation or an object. The handle destructor insures that the underlying proxy is dereferenced.

Detailed Description Text (3467):
Globally Addressable Interface--This is a pattern for making interfaces publicly available. Distributed connections to Globally Addressable Interfaces can be pooled using the Refreshable Proxy Pooling pattern.

Detailed Description Text (3468):
Proxy--This pattern is documented in the book "Design Patterns" by Gamma, Helm, Johnson

and Vlissides. The proxy pattern is often used to communicate with server components in a distributed environment. Proxies are pooled using the Refreshable Proxy Pool pattern.

Detailed Description Text (3469):
Trader--The Trader service defines how distributed architectures locate components based on the types of services they provide. The allocation queue interacts with a Trader Service to allocate the correct type of proxy.

Detailed Description Text (3470):
Naming--The Naming Service provides a mapping between names and object references. A Naming Service could be used to store the GAI references that the Refreshable Proxy Pool pattern requires.

Detailed Description Text (3472):
Single Use--As opposed to pooling connections to a remote server a client can request a new connection each time a GAI is needed. This would work best when a client infrequently needs GAIs.

Detailed Description Text (3473):
Proxy Pool--This pattern addresses the pooling of proxies without periodic refreshing. It is a simpler version of the Refreshable Proxy Pool that may be of use when server load is fairly constant.

Detailed Description Text (3557):
The data structure may be bundled on the server by a business object. In addition, the business object may be instantiated by an action of the client. Also, the network may be at least one of a local area network and a wide area network. As a further option, the request may be administered by a proxy component. Further, the data structure may contain no logic.

Detailed Description Text (3568):
The Client would first determine the sum total of everything it will need from the business object on the Server machine. The Client makes a request for all of this data from the business object. The business object bundles all the data into a data structure and returns it to the client. The Client will cache this data (using the Caching Proxy pattern) on its local client machine and use it as needed.

Detailed Description Text (3576):
1. The client instantiates a Proxy (Customer Component Proxy) to the Customer Component. The Client then asks the Proxy for Customer Jimbo Jones.

Detailed Description Text (3577):
2. The Customer Component Proxy forwards the request across the network to the Customer Component.

Detailed Description Text (3582):
7. The Client creates a proxy to the "Jimbo Jones" object using the remote object reference.

Detailed Description Text (3583):
Now that a customer object (Jimbo Jones) exists on the server component, Structure Based Communication can be used to pass the needed data from the server to the client.

Detailed Description Text (3586):
8. The Client asks the Customer Proxy for the data associated with the Jimbo Jones object.

Detailed Description Text (3587):
9. The Customer Proxy forwards the request across the network to the Customer Component.

Detailed Description Text (3589):
11. The Data Structure is passed across the network to the Customer Proxy on the Client.

Detailed Description Text (3590):
12. The Customer Proxy forwards the data structure containing Jimbo Jones' data to the Client component.

Detailed Description Text (3595):
Customer Component Proxy--A proxy to the Customer Component. Any request it receives, it forwards across the network to the Customer Component.

Detailed Description Text (3596):
Customer Proxy--A proxy to the Jimbo Jones Customer Object Any request it receives, it forwards across the network to the Jimbo Jones Customer Object.

Detailed Description Text (3603):
The following block of code is the "main" routine for the Client. Even though all distributed calls are made through a proxy, the proxy code is not shown in this example. The two proxy classes encapsulate the details associated with distributed network calls.

Detailed Description Text (3606):
Note: The "Main" code example above obtains a Proxy to customer "Jimbo Jones" and then a Structure of data through the proxy. The code was written for ease of understanding, but causes two calls across the network. In reality, it is preferred to perform both functions using one network call. Limiting the number of network calls will improve system performance. The recommended code might look something like this:

Detailed Description Text (3608):
Proxy--This pattern is documented in the book "Design Patterns" by Gamma, Helm, Johnson and Vlissides. The proxy pattern is often used to communicate with server components in a distributed environment. The Proxy pattern is often used to retrieve data structures from a server component.

Detailed Description Text (3609):
Cached Proxy--This pattern is documented in "The Proxy Design Pattern Revisited" section of the Pattern Languages of Programming Design 2 book. A Cached Proxy caches data locally on the client. Structure Based Communication uses and builds upon this pattern.

Detailed Description Text (3611):
Locally Addressable Interface--This pattern can also be used in conjunction with Structure Based Communication. After establishing a relationship with an LAI, a client may obtain data from the Server object using Structure Based Communication.

Detailed Description Text (3622):
A plurality of presentation interfaces may be interfaced. Optionally, a model may be interfaced for management purposes. With such an option, the model may further include a proxy. As another option, errors and exceptions may also be handled. As a third option, events intended to be received may be triggered by the presentation interface.

Detailed Description Text (3627):
The system may also require a number of interfaces to complete the same use case. Depending on the user category executing the use case, the interface may be a PC, a handheld device, or a telephone. Even on the same type of device, the interface may differ depending on the user category. Some users may want to access the application via a standard Windows interface while others may want to access it via a "web-centric" interface (internet browser). In all of these cases, the unit of work to be completed by the user is not changed and should be reused.

Detailed Description Text (3649):
The diagram shows the various "layers" of interaction where the lightest shaded boxes are the presentation, the next darkest is the activity, and the darkest is the component (proxies 12700).

Detailed Description Text (3789):
FIG. 141 illustrates a GarbageCollector 14100 requesting for interest in context B 14102. Client 2 registers interest so the reaper updates the access time stamp and maintains B.

Detailed Description Text (4044):
The mapper manager is responsible for creating mappers for a given CLASS_ID. Each type of mapper registers with the mapper factory when the shared object is loaded into memory (using the dynamic registration factory pattern). The mapper factory is a singleton read only object.

Detailed Description Text (4243):
FIG. 179 illustrates a flowchart for a method 17900 for allowing a batched request to indicate that it depends on the response to another request. A group of business objects necessary for a transaction are provided in operation 17902. Logically-related requests received from the business objects are batched into a single network message in operation 17904. One of the requests is a parent request. Received from the parent request in operation 17906 is a register that at least one of the requests is dependent upon the response data. The network message is sent across a network and the requests are unbundled from the network message in operations 17908 and 17910. Upon receipt of a response to the parent request in operation 17912, data is directed from the response to the parent request to the dependent request in operation 17914. Received from the response to the parent request is a response to the dependent request based on the received data in operation 17916.

Detailed Description Text (4253):
Therefore, additional mechanisms should allow a batched request to indicate that it depends on another request. If a business object does not have its primary key--or other attributes guaranteeing a unique match--it becomes a Dependent Request. Because a dependent cannot fetch itself, some other business object will inevitably have the necessary foreign key. The dependent request will therefore register its dependency on this other object.

Detailed Description Text (4256):
Dependencies should not be hard-coded. Any business object can register as dependent on any other object which can provide the necessary data.

Detailed Description Text (4257):
In this manner, dependencies can be set at run-time. This could happen while building the model, or as requests register with Request Batcher.

Detailed Description Text (4261):
Loose Coupling. When a request dynamically registers its dependency, it need not know anything about its parent request. The dependent effectively says, "I don't know who you are, but I know that your response data contains my identifier."

Detailed Description Text (4293):
The LUW Context holds onto all domain objects in a particular model. It can therefore collaborate with an Identity Registration Manager, to enforce object uniqueness within the particular context.

Detailed Description Text (4305):
FIG. 187 illsutrates a manner in which the present invention registers requests.

Detailed Description Text (4306):
A Request Batcher 18700 object will group logically-related requests. All requests will register with this coordinating object, rather than sending themselves immediately and independently to their server or database. The batcher will then store these requests together, until told to send them as a unit. This batching applies equally well to update and retrieval transactions.

Detailed Description Text (4311):
Scaleability. In an asynchronous or multi-threaded environment, an application could use multiple Request Batchers. For example, each LUW could have its own batcher. A batcher needs to store state while building the batch, as requests register. Using multiple instances facilitates registration for, and sending of, multiple batches simultaneously. Users can then multi-task while other, time-consuming requests process in the background.

Detailed Description Text (4312):
This simultaneity can also be supported with one, multi-threaded batcher. In this case, each request registers along with its unique transaction id.

Detailed Description Text (4328):
However, with Individual Persistence, a transaction no longer maps to a centralized module. Instead, independent requests register for the transaction in an ad-hoc manner. FIG. 189 illustrates an Ad Hoc Registration feature.

Detailed Description Text (4331):

Multiple requests can no longer send themselves directly to the server, in an ad hoc fashion. Instead, they must register with a centralized object, which can sort them first. A centralized Request Sorter will order multiple requests before finally sending the transaction.

Detailed Description Paragraph Table (1):
sample protocol functionality options architecture service controlling media RTSP or proprietary Streaming Messaging delivery service monitoring data RTCP or proprietary Streaming Messaging stream service end-to-end delivery RTP or proprietary Streaming Messaging of stream service message transport UDP, Multicast UDP, Message Transport TCP service packet IP, IP Multicast Packet forwarding/internet- Forwarding/Internet-working working service

Detailed Description Paragraph Table (21):
Class Description PooledProxy (10402) This is the base class for the pooled proxy. It actually acts as a wrapper for a Proxy and maintains all usage and reference counting information. ProxyPool (10404) This is the proxy pool, where clients go to retrieve a proxy. It should be thread-safe in that multiple threads are automatically synchronized. This pool should only contain valid proxies that have been allocated by the AllocationPool. When a proxy is requested, the usage count is incremented. After the "usage" passes retirement age, the proxy is remove from the pool and placed back into the allocation pool. AllocationPool (10406) This is the pool that actually does the proxy allocation. This pool is populated with unallocated proxies and a "reader" thread will allocate them. Since there will only be one, this class should be implemented as a Singleton. This pool however can allocate proxies of any type. ProxyHandle<T> (10408) This is the Handle that clients should use to manage pooled proxies. The handles must use a static_cast<T> (C++ template) to retrieve the correct proxy. T is defined by a client template instantiation, and assumes the client knows exactly what type of proxy the pool is actually holding. Clients must take care to assure that pools only contain proxies of one type.

Detailed Description Paragraph Table (27):
// Client side code here. Main() { // Create a Proxy to the Customer Component. CustomerComponentProxy aCustomerComponentProxy = new CustomerComponentProxy(); // Get a Proxy to the Jimbo Jones Customer // object. CustomerProxy aCustomerProxy = aCustomerComponentProxy.getCustomer("Jimbo Jones"); // Get Customer data from the Customer Server // Component(Call across the network) CustomerStructure aCustomerStructure = aCustomerProxy.getCustomerAsStructure(); // Use the Customer data received in the // structure. For Example, display the data // structure data (aCustomerStructure) in a UI. . . }

Detailed Description Paragraph Table (30):
// Create a Proxy to the Customer Component // (same as above) CustomerComponentProxy aCustomerComponentProxy = new CustomerComponentProxy(); // Get a Proxy to the Jimbo Jones Customer // object // AND // customer data at the same time. CustomerProxy aCustomerProxy; CustomerStructure aCustomerStructure = aCustomerComponentProxy.getCustomerData("Jimbo Jones", aCustomerProxy);

Detailed Description Paragraph Table (43):
interface LoadBalancer { Object get Service ( ) raises (ArchitectureException); void register (in Object aServerComponent) raises (ArchitectureExeption); };

L10: Entry 36 of 39 File: USPT Feb 13, 2001

DOCUMENT-IDENTIFIER: US 6189096 B1
TITLE: User authentification using a virtual private key

Abstract Text (1):
A method, computer system, and program product provides for authentication of user messages using PKI technology in environments where limited capacity prevents direct PKI technology use, and strong security is provided using magnetic swipe cards or the like, and a pass phrase is used for enhanced security and to avoid the need for special purpose devices. The invention is advantageous where there are limitations on the space available for PKI credentials, such as in the userid and password fields of a remote access protocol. PKI techniques are used without transferring lengthy keys or certificates once an initial registration process is complete. A secret key is used. A digest is computed of the secret key, the user's certificate serial number, and a time stamp. The digest, together with the user's certificate serial number and the time stamp, forms a compact message that may be transmitted. Private keys and secret keys are not sent during authentication. Replay attacks are prevented.

Brief Summary Text (13):
Another example of a limiting environment exists in remote access systems. Here, the client station does not communicate directly with a security server. Instead, the client station communicates with a communications server, which, in turn, communicates with a remote access security server. The protocol used for communication between the client station and the communications server is typically designed to get a userid and password from the user. A typical example of such a protocol is the Point to Point Protocol (PPP). Such userid/password oriented protocols can pass about 60 bytes in their userid/password fields, which is insufficient to support for the direct use of public key technology for user authentication, encryption, or for digital signatures. Thus, PKI authentication cannot effectively be used in this type of remote access system.

Brief Summary Text (22):
This invention involves solving the above-identified problems using digests in a two step process of registration and authentication.

Brief Summary Text (23):
In one preferred embodiment, there is a method of user authentication using PKI technology in environments where limited capacity prevents direct PKI technology use. In a magnetic swipe card system, the data storage is the capacity that is limited. In a remote access (dial-up) system, the length of the userid/password fields is the capacity that is limited. The method according to the invention is most useful where there are limitations on the space available for PKI credentials.

Brief Summary Text (25):
In the case of applying the method of the invention to remote access environments, the invention modifies both the conventional registration and authentication processes normally used.

Brief Summary Text (27):
In the main, the invention resides in a method, a computer system, and a computer program product providing for authentication of user messages using PKI technology in environments where limited capacity prevents direct PKI technology use. The invention is advantageous where there are limitations on the space available for PKI credentials, such as in the userid and password fields of a remote access protocol. PKI techniques are used without actually transferring lengthy keys, certificates, or digital signatures once an initial registration process is complete. A private key authenticates a user at a client and is used to retrieve a stored, encrypted secret key. A digest is computed of the secret key, the user's X.509 ISO standard public key

certificate, and a time stamp. To further minimize the size of the message, the unique serial number of the user's certificate (the certificate serial number, also referred to as the certificate s/n) may be employed. The digest, together with the user's certificate serial number and the time stamp, forms a compact message that may be transmitted in the userid and password fields of a remote access protocol. The private key and the secret key are not sent. The secret key, stored beforehand at the server, is used along with the sent user's certificate serial number and the sent time stamp to compute another digest which is compared with the first digest. When the two digests match, the user is considered authentic. The time stamp is used to prevent replay attacks.

Brief Summary Text (28):
In a second embodiment of the invention, there is provided a way to use certain information referred to as a "reference" instead of a user's private key. Basically, the second embodiment differs from the first embodiment in that the user's private key is required during only the registration process. Thereafter, the user's private key is not used but, rather, a reference is read from something the user has, such as a magnetic swipe card or a biometric device. The reference is digested to provide a client secret key, and a preliminary digest is made of the user's certificate serial number, a time stamp, and this secret key. This preliminary digest is sent, along with the user's certificate serial number and the time stamp, to the authentication server. The authentication server may store the reference itself or may store a digested version of the reference. The digested reference serves as the server secret key. Upon receipt of the message, authentication is performed by digesting the time stamp and user certificate serial number and secret key, and comparing this computed digest with the preliminary digest sent in the message. This embodiment of the invention is advantageous in that the reference is not stored at the client. A hacker cannot obtain the reference by attacking the client station. Also, the user's private key is not used after registration. Moreover, when the user has a magnetic swipe card or the like, the user can very easily determine when the card is missing. Instead of a magnetic swipe card, the reference may be provided by a fingerprint reader, retinal scanner, or the like. In addition, the reference itself is sent only during the registration process, and thereafter is not per se sent over the network.

Brief Summary Text (29):
According to the third preferred embodiment of the invention, there is provided a passphrase which substitutes for the reference. In other words, the third embodiment is substantially similar to the second except that the user does not provide a "thing" such as a swipe card or a fingerprint. The user provides from memory a passphrase which serves as a reference. Like the reference, the pass phrase is not stored at the client and cannot therefore be discovered by hacker. As in the second embodiment, the user's private key is used during only the registration process, and the passphrase is not per se sent over the network afterward. Moreover, the third embodiment of the invention does not require any card reader or biometric device because the pass phrase may be entered using a keyboard.

Drawing Description Text (10):
FIG. 8 illustrates a registration procedure for a Virtual Private Key (VPK).

Drawing Description Text (12):
FIG. 10 illustrates a registration procedure in different embodiments of the invention.

Drawing Description Text (13):
FIG. 11 illustrates an authentication procedure corresponding to the registration procedures of FIG. 10.

Detailed Description Text (5):
Security is a serious problem on the Internet and other public networks today. An important aspect of network security is user authentication. User authentication includes the verification of the identity of a user at the initiation of a session or other activity, and also the prevention of unauthorized mimicry of an already-verified user.

Detailed Description Text (6):
To deal with security, the industry has adopted a security server approach. Under this approach, a security server is interposed between a client and an applications server. The role of the security server is to be the sole link between the client and the applications server. The security server establishes communications between the client

and the applications server if and only if the user at the client is authenticated. The term "security server," as used in this sense, is meant to encompass security servers, proxy servers, firewalls, and authentication servers.

Detailed Description Text (14):
One knowledgeable in computer systems will appreciate that "media", or "computer-readable media", as used here, may include a diskette, a tape, a compact disc, an integrated circuit, a cartridge, a remote transmission via a communications circuit, or any other similar medium useable by computers. For example, to supply software that defines a process, the supplier might provide a diskette or might transmit the software in some form via satellite transmission, via a direct telephone link, or via the Internet.

Detailed Description Text (45):
Internet RFC 1492, "An Access Control Protocol, Sometimes called TACACS", July, 1993 (see http://leviathan.tamu.edu:70/1s/internet/rfcs); and

Detailed Description Text (46):
Internet RFC 2138, "TXT Remote Authentication Dial in User Service (RADIUS), January, 1997 (see http://leviathan.tamu.edu:70/1s/internet/rfcs).

Detailed Description Text (47):
Registration

Detailed Description Text (48):
The invention involves an authentication procedure which must be preceded by a registration procedure. The registration procedure will first be described with reference to FIG. 8, in which client 110, security server 120, and communications server 700 are shown.

Detailed Description Text (49):
First, a user, via client 110, requests of the communications server 700 a connection with security server 120. The communications server notifies the security server of this requested connection, and then solicits the user for his userid and password. Assuming that a valid userid and password are supplied at this point, the communications server establishes a connection between client 110 and security server 120. Once this connection is established, the client 110 and the security server 120 can communicate and send messages of arbitrary length and the communications server 700 is of no particular import. The foregoing actions are represented as step 810.

Detailed Description Text (57):
The registration procedure need be performed only one time, but must precede the first operation requiring the authentication procedure to be performed. It will be understood that, in the preferred embodiment of the invention, a security server registers many users. As users (and also their respective secret keys) are registered, they may be thought of as being among a plurality of registered users. Their secret keys may be thought of as being among a plurality of registered keys.

Detailed Description Text (59):
The registration procedure having been described and illustrated, a description will now be provided of the authentication procedure.

Detailed Description Text (66):
Not shown in FIG. 9 are the steps of the client contacting the communications server and requesting a connection with the security server; the communications server contacting the security server; and the communications server soliciting the user's userid and password from the client.

Detailed Description Text (68):
In step 970, the security server takes the transmission message data that was supplied in the userid and password fields and extracts its constituent parts, namely, the user's certificate serial number, the time stamp determined in step 930, and the preliminary digest computed by the client in step 940. In step 975, the security server uses the user's certificate serial number to retrieve from storage (i.e., from the authentication table) the corresponding user's certificate and encrypted secret key, and decrypts the secret key with its own private key. Although the term "retrieve" is used here, it is quite possible that the corresponding secret key already happens to be in RAM or the like. Thus, retrieve may simply mean moving the proper key information in to rapid is access registers as necessary. "Retrieve" should not be construed to be

limited to an access of a disk.

Detailed Description Text (71):
The step 990 considers as invalid messages that do not have a time stamp that is later than the most recently received message. One way to express this is to say that a present message is received, and its time stamp is stored. Then, a subsequent message is received, and its time stamp is compared to the stored time stamp of the message that was, until receipt of the subsequent message, the present message. Here, it should be noted that t he term "stored" does not mean only storage to a location on a disk, for example. The essence of this term in this context is that the value of the previous message time stamp is remembered in some manner, whether written on disk, saved in RAM, or written into a register.

Detailed Description Text (74):
The secret key is passed only once, in encrypted form, from the client to the security server. This occurs during the registration procedure. The secret key is used, however, every time the authentication procedure is carried out. It is needed at the client and at the security server. Thus, the secret key may be referred to as a Virtual Private Key (VPK) because it is used so frequently but never actually passed during the authentication process.

Detailed Description Text (75):
The foregoing registration procedure and authentication procedures amount to a method which allows full public key benefits without the actual transmission of the relatively long PKI credentials. Because the transmission message with its preliminary digest is quite short, it can be sent in short data fields. It may be said that the above-identified invention provides a way to map relatively long PKI information onto short data fields.

Detailed Description Text (80):
Registration

Detailed Description Text (81):
The registration procedure will be described with reference to FIG. 10. Communications server 700 is not shown.

Detailed Description Text (82):
In step 810, a user, via client 110, requests a connection with security server 120. A connection is established between client 110 and security server 120.

Detailed Description Text (90):
Again, the registration procedure need be performed only one time, but must precede the first operation requiring the authentication procedure to be performed.

Detailed Description Text (92):
It is to be noted that the user's private key used to prepare the registration will not be required for subsequent authentication operations and therefore the magnetic swipe card or the like will be thereafter sufficient.

Detailed Description Text (101):
In step 980, the security server computes a digest using the user's certificate serial number (provided in the transmission message), the time stamp (also provided in the transmission message), and the secret key (retrieved from the security server's own storage) . In step 985, the security server compares the just-computed digest with the reliminary digest that was provided in the transmission message. The two digests can be equal only when the sender of the transmission message possesses a magnetic card with contents that, when digested, match the previously sent user's secret key. Since the user's secret key and reference are not stored at the client, this further implies that the sender possesses the magnetic card that was used at the time of registration.

Detailed Description Text (104):
The secret key is passed only once, in encrypted form, from the client to the security server. This occurs during the registration procedure. The secret key is used, however, every time the authentication procedure is carried out. The secret key is a Virtual Private Key (VPK); it is frequently used but never actually passed during authentication.

Detailed Description Text (109):
Registration

Detailed Description Text (110):
The registration procedure will be described with reference to FIG. 10. It is to be understood that this third embodiment differs from the second embodiment only in that the reference is provided from a user's memory instead of via some device.

Detailed Description Text (112):
In step 810, a user, via client 110, requests a connection with security server 120. A connection is established between client 110 and security server 120.

Detailed Description Text (116):
The registration procedure need be performed only one time, but must precede the first operation requiring the authentication procedure to be performed.

Detailed Description Text (121):
The secret key is passed only once, in encrypted form, from the client to the security server. This occurs during the registration procedure. The secret key is used, however, every time the authentication procedure is carried out. The secret key is a Virtual Private Key (VPK); it is frequently used but never actually passed during authentication.

CLAIMS:

2. The method of authentication as set forth in claim 1, further comprising:

a registration step which comprises:

generating said client secret key;

encrypting said client secret key with a public key of said user to provide said encrypted client secret key value;

providing a secret key message, encrypted with a server public key, said secret key message including said client secret key, said user certificate serial number, and a user digital signature;

performing decryption of said encrypted said secret key message using said server private key;

determining said secret key message to be authentic based on said decryption being successful and based on said digital signature; and

storing said client secret key as said server secret key in association with said user certificate serial number.

10. The method for server authentication as set forth in claim 9, wherein said step of storing includes:

receiving a registration request message from a user;

sending to said user a server certificate;

receiving a secret key message;

decrypting said secret key message using a public key of said user to provide a decrypted secret key message;

obtaining a digital signature and a sent secret key from said decrypted secret key message;

authenticating said secret key message based on said digital signature; and

storing at said server said sent secret key as one of said plurality of registered secret keys.

13. The method for server authentication as set forth in claim 12, wherein said step of storing includes:

receiving a registration request message from a user;

sending to said user a server certificate;

receiving a secret key message;

decrypting said secret key message using a public key of said user to provide a decrypted secret key message;

obtaining a digital signature and a sent secret key from said decrypted secret key message;

authenticating said secret key message based on said digital signature; and

storing at said server said sent secret key as one of said plurality of registered secret keys.

32. The computer system adapted to authenticate messages, as set forth in claim 31, wherein said memory further includes software instructions adapted to enable the computer system further to perform said step of storing said plurality of registered secret keys by:

receiving a registration request message from a user;

sending to said user a server certificate;

receiving a secret key message;

decrypting said secret key message using a public key of said user to provide a decrypted secret key message;

obtaining a digital signature and a sent secret key from said decrypted secret key message;

authenticating said secret key message based on said digital signature; and

storing said sent secret key as one of said plurality of registered secret keys.

35. The computer system adapted to authenticate messages, as set forth in claim 34, wherein said memory further includes software instructions adapted to enable the computer system further to perform said step of storing said plurality of registered secret keys by:

receiving a registration request message from a user;

sending to said user a server certificate;

receiving a secret key message;

decrypting said secret key message using a public key of said user to provide a decrypted secret key message;

obtaining a digital signature and a sent secret key from said decrypted secret key message;

authenticating said secret key message based on said digital signature; and

storing said sent secret key as one of said plurality of registered secret keys.

48. The computer program product for enabling a computer to authenticate messages, as set forth in claim 47, wherein:

said software instructions further enable the computer to perform said predetermined operations including:

receiving a registration request message from a user;

sending to said user a server certificate;

receiving a secret key message;

decrypting said secret key message using a public key of said user to provide a decrypted secret key message;

obtaining a digital signature and a sent secret key from said decrypted secret key message;

authenticating said secret key message based on said digital signature; and

storing said sent secret key as one of said plurality of registered secret keys.

51. The computer program product for enabling a computer to authenticate messages, as set forth in claim 50, wherein:

said software instructions further enable the computer to perform said predetermined operations so that said step of storing said plurality of registered secret keys includes:

receiving a registration request message from a user;

sending to said user a server certificate;

receiving a secret key message;

decrypting said secret key message using a public key of said user to provide a decrypted secret key message;

obtaining a digital signature and a sent secret key from said decrypted secret key message;

authenticating said secret key message based on said digital signature; and

storing said sent secret key as one of said plurality of registered secret keys.

WEST

Generate Collection

L13: Entry 2 of 5 File: PGPB Feb 7, 2002

DOCUMENT-IDENTIFIER: US 20020016912 A1
TITLE: System and computer based method to automatically archive and retrieve encrypted remote client data files

Abstract Paragraph (1):
A remote user establishes an interactive session. A pre-determined backup set is encrypted at a remote user site according to a key based upon the user's password. Encrypted backup data is then transmitted to a backup archive server and decrypted utilizing the user's password generated key and re-encrypted according to a user specified backup set key and stored upon an auxiliary storage unit. The present invention further embodies a retrieval process wherein a remote user designates one or more files to be retrieved and the backup set encryption key used during the initial backup operation. Software executing within the backup archive server CPU retrieves and decrypts the specified files according to the originally specified backup set key and re-encrypts the files according to the remote user's password. Retrieved information is then transmitted to the remote user whereupon it is subsequently decrypted at the remote user site.

Cross Reference to Related Applications Paragraph (1):
[0001] This application is a continuation-in-part of (a) U.S. patent application Ser. Nos. 09/239,425 entitled "A Secure Electronic Mail System" filed on Jan. 28, 1999 and (b) 09/255,837 entitled "Method For Information Encoding And Transfer" filed on Feb. 23, 1999 which are continuation-in-part applications of U.S. patent application Ser. No. 08/892,982, filed Jul. 15, 1997, and entitled "Combined Remote Access and Security System", now U.S. Pat. No. 5,970,149; which is a continuation-in-part of U.S. patent application Ser. No. 08/752,249, filed Nov. 19, 1996, and entitled "Combined Remote Access and Security System".

Summary of Invention Paragraph (6):
[0004] The present invention provides for secured utilization of an encrypted backup archive. The instant invention advances the art by allowing its practice to be supported via an encrypted communications protocol interfacing with, and relying upon, the teachings, practices and claims disclosed in co-pending U.S. patent application Ser. Nos. 09/239,425 and 09/255,837 (hereinafter synonymously referred to as "Secure Agent" or "SA").

Summary of Invention Paragraph (7):
[0005] Secure Agent Service Overview

Summary of Invention Paragraph (8):
[0006] The following overview is provided to facilitate a comprehensive understanding of the teachings of the instant invention. Secure Agent utilizes a secure login sequence wherein a client connects to a Secure Agent server using a key known to both systems and a client connects and presents the server with user identification (as used herein the term "client" refers synonymously to a remote user establishing, and communicating with the instant invention through Secure Agent allocation and encryption processes as taught in the above noted applications). If recognized, the Secure Agent server initiates a protocol whereby the client's identification is verified and subsequent communication is conducted within a secured (encrypted) construct. For purposes of this overview, the term "server" should be considered a hardware configuration represented as a central processing unit wherein Secure Agent and Host DLL's are executed. The term "DLL" as used herein refers to a Secure Agent host dynamically linked library (a.k.a. Host DLL). The term "DLL" or "dynamically linked library" is used in a manner consistent with that known to those skilled in the art. Specifically, the term "DLL" refers to a library of executable functions or data that can be used by a Windows application. As such, the instant invention provides for one or more particular functions and program access to such functions by creating a static

or dynamic link to the DLL of reference, with "static links" remaining constant during program execution and "dynamic links" created by the program as needed.

Summary of Invention Paragraph (9):
[0007] The Secure Agent server presents a variable unit of data, such as the time of day, to the client as a challenge. The client must then encrypt that data and supply it back to the server. If the server is able to decrypt the data using the stored client's key so that the result matches the original unencrypted challenge data, the user is considered authenticated and the connection continue. The key is never passed between the two systems and is therefore never at risk of exposure.

Summary of Invention Paragraph (11):
[0009] The access rights of each client (what the client is able to accomplish during a session) is governed by data stored on the Secure Agent server to which the client is associated. As an example, such rights might encompass the ability to administer and utilize the services of the server system, which would, in turn, include capabilities such as adding new client users, changing a user's rights, loading new code to the server, using a feature (or service) of the server and more.

Summary of Invention Paragraph (12):
[0010] Consequently, Secure Agent allows for the transmission of new code to the server and for that code to be implemented upon demand by a client. Such dynamic, real-time implementation in turn, allows for the behavior of the server to be modified. It is to this behavior modification the instant invention addresses its teachings, and thereby advances the contemporary art.

Summary of Invention Paragraph (13):
[0011] As will be readily appreciated by those skilled in the art, though the instant invention utilizes encryption/decryption and code recognition technology associated with Secure Agent, an alternative technology may be employed in support of the instant invention without departing from the disclosure, teachings and claims presented herein.

Summary of Invention Paragraph (15):
[0012] The present invention is best viewed as comprised of at least two server software components with one or more client subcomponents or sub-processes disclosed in association thereto. As used throughout the instant invention specification and claims, the term "server" is used synonymously with "server central processing unit", "server CPU", and the term "client" is used synonymously with "host user", "client central processing unit", "client CPU" and "remote user".

Summary of Invention Paragraph (16):
[0013] It is an object of the instant invention to provide for one or more remote backup archive server Central Processing Units (CPU) to backup one or more remote user files.

Summary of Invention Paragraph (23):
[0020] Another object of the instant invention is to provide a system wherein backup data files encrypted according to a user password key are received from a remote user at a backup archive server then decrypted using said encryption key based on the user's password and re-encrypted prior to storage using a backup set encryption key specified by the user during file backup.

Detail Description Paragraph (7):
[0035] Also communicably attached to the backup archive server central processing unit 3 are one or more remote users 7 (herein synomously referred to as clients, remote clients, users) each shown executing under the dispatching control of their respective operating systems first software 10. It is herein noted that the communicable attachment of central processing units in FIG. 1 may be facilitated via any network compatible means over which digitized information may be transmitted. Client side central processing unit configurations are compatible with those typically employed as web servers, database servers, application servers, workstations, personal computers, desktop computers and laptop computers. Backup archive server central processing units 3 of the instant invention are typically those hardware central processing unit configurations capable of providing information and services to one or more remote users such as, but not limited to, web servers, database servers and application servers. Said backup archive server central processing unit 3 may alternatively be configured to accommodate a single or multiple control programs each of which is capable of responding to remote user requests and interaction requirements. The

resiliency and flexibility of the instant invention, however, allows for one or more backup archive server processing units 3 to be embodied as a mainframe computer employing one or more control programs each of which may have under its dispatching control functionality of multiple second 9 and third software 11 instruction sets.

Detail Description Paragraph (9):
[0037] In FIG. 2, a remote user first defines a backup set as seen at box 22 indicating information he/she desires to be stored upon the auxiliary storage device 5 of the instant invention's backup archive server central processing unit 3. To provide parameters for said backup set, the user specifies those directories and/or files to be backed up or to be excluded from said backup operation as seen at box 24, the frequency and time of day at which the specified files are to be backed up as seen at box 26, an encryption key for the backup data set as seen at box 28 and the identification of the backup archive server CPU he/she wishes to use 29. It is emphasized that the instant invention affords the remote user the security of encrypting files according to a logon password key with encrypted files received from the remote user at a backup archive server then decrypted using this password based encryption key and re-encrypted prior to storage using a backup set encryption key specified by the user during file backup and retrieval processes. Automated password key based encryption is facilitated via a secure login and transmission protocol such as disclosed in detail in co-pending U.S. patent application Ser. No. 09/239,425 entitled "A Secure Electronic Mail System" filed on Jan. 28, 1999.

Detail Description Paragraph (10):
[0038] The co-pending application further includes references to supporting detail and disclosure with said detail, disclosure and co-pending application incorporated herein by reference in their totality. Once the remote user 7 has logged on to the backup archive server 3 and been verified via second software 9 as an authorized user, first software 10 operating within each remote user central processing unit 7 encrypts data to be transmitted to the backup central processing unit according to a generated encryption key based upon the user's password. As scheduled, or optionally via a manual input entry, the remote client machine 7 logs onto the backup archive server 3 using a preauthorized user id and password 30.

Detail Description Paragraph (11):
[0039] The backup of specified user files is then initiated via transmission of the encrypted data stream encoded according to the user's password key 32. In so doing, the first software 10 of the instant invention ensures that only files which have been modified since the last backup are transmitted to the backup archive server central processing unit 3. User files are then decrypted via third software 11 using the remote user's password encryption key and then re-encrypted according to a remote user determined and dynamically specified backup key 36, said backup key specified in element 28. The re-encrypted information is then stored in an auxiliary storage device as indicated earlier, such as a mass storage unit 5 or optional CD burner 6.

Detail Description Paragraph (12):
[0040] FIG. 3 illustrates the invention's computer based method for verifying, facilitating and retrieving backed up remote client data files. In FIG. 3, a remote user first logs onto a backup archive server 3 using a predefined user id and password as shown at box 42. The successfully logged on user then specifies the location where retrieved files are to be stored on the remote/client machine as shown at box 44, and then specifies a location from which the remote user wishes to retrieve files from as shown at box 46. Having specified the location from which to recover retrievable and selective backup files, the user next specifies the encryption key of the backup set used when the backup set was created 48. The encryption key had been designated initially at the time of the backup sets' creation.

Detail Description Paragraph (13):
[0041] The user next determines which files are to be recovered as shown at box 50 and third software 11 residing within the backup archive server central processing unit 3 retrieves those files and decrypts them on the server as shown at box 52, utilizing the client provided encryption key for decrypting purposes. The decrypted retrieved files are then re-encrypted according to the remote user's sign on/password based encryption key as shown at box 54 and the re-encrypted files are then transmitted to the remote user as shown at box 56. Once received by the remote user, the remote user once again via his/her based password encryption key decrypts the files and renders recognizable data contained therein.

CLAIMS:

1. A system to automatically archive and retrieve encrypted remote client data files comprising: at least one security administrator central processing unit communicably attached to at least one backup archive server central processing unit; at least one auxiliary storage device communicably attached to said backup archive server central processing unit; at least one client central processing unit communicably attached to said backup archive server central processing unit; first software for transmitting at least one remote client backup file encrypted according to the remote client's password from said client central processing unit to said backup archive server central processing unit and for processing encrypted data communicated from second software, said first software executing within said client central processing encrypted unit; second software for processing encrypted data communicated from said first software, validating authorized remote client access and storing said transmitted client back-up file, said second software executing within said backup archive server central processing unit; and third software for (a) identifying remote client data which has been encrypted according to a remote user password key; (b) decrypting the data identified in (a); (c) re-encrypting the data decrypted in (b) ; according to a remote user determined and specified backup set key; (d) storing data re-encrypted in (c) to an auxiliary storage device; (e) identifying and retrieving remote client data which has been encrypted according to a remote user determined and specified backup set key; (f) decrypting the data identified in (a); (g) re-encrypting the data decrypted in (b) according to a remote user password key; and, (h) transmitting data re-encrypted in (c) to a remote user authorized to receive said data.

7. A computer based method for automatically archiving and retrieving remote client data files comprising: (a). a remote user specifying a back-up set to archive; (b). said remote user specifying to be archived or excluded; (c). said remote user specifying the frequency and time of day of automated by archiving activity; (d). said remote user specifying an encryption key for said defined backup set; (e). said remote user scheduling or manually requesting a remote client machine log on to a backup archive server; (f). said user specifying the identification of a backup server CPU to be utilized for backup purposes; (g). transmitting to a backup archive server remote user defined files which have been modified since last backup; (h) decrypting at said backup archive server data identified in (g); (i) re-encrypting the data decrypted in (h) according to the remote user determined and specified backup set key in (d); and (j) storing data re-encrypted in (i) to an auxiliary storage device.

8. The computer based method for automatically archiving and retrieving remote client data files according to claim 7 further comprising: (a) a remote user logging on to a backup archive server using a pre-authorized user ID and password; (b). said remote user specifying a client CPU file location to which retrieved files are to be stored; (c). said remote user specifying an appropriate backup set from which to recover files; (d). said remote user specifying the encryption key of the backup set to be restored; (e). said remote user specifying files within said backup set to be retrieved; (f). retrieving remote client data which has been encrypted according to a remote user determined and specified backup set key; (g). decrypting the data retrieved in (f); (h). re-encrypting the data decrypted in (g) according to said remote user's password key; and, (i). transmitting data re-encrypted in (h) to a remote user authorized to receive said data.

9. A computer based method for automatically archiving and retrieving remote client data files comprising: (a). a remote user specifying a back-up set to archive; (b). said remote user specifying files to be archived or excluded; (c). said remote user specifying the frequency and time of day of automated archiving activity; (d). a remote user specifying an encryption key for said defined backup set; (e). said remote user scheduling or manually requesting a remote client machine log on to a backup archive server; (f). said user specifying the identification of a backup server CPU to be utilized for backup purposes; (g). transmitting to a backup archive server remote user defined files which have been modified since last backup; (h). decrypting the data identified in (g); (i). re-encrypting the data decrypted in (h) according to the remote user determined and specified backup set key in (d); (j). storing data re-encrypted in (i) to an auxiliary storage device; (k). said remote user identifying a remote client CPU file location to which retrieved files are to be stored; (l). said remote user identifying an appropriate backup set from which to recover files; (m). said remote user identifying the encryption key of the backup set to be restored; (n). said remote user identifying files within said backup set to be retrieved; (o). retrieving remote client data which has been encrypted according to said remote user determined and specified backup set key; (p). decrypting the data retrieved in (o); (q) re-encrypting the data decrypted in (p) according to said remote user's password key; and, (r)

transmitting data re-encrypted in (q) to a remote user authorized to receive said data.

L13: Entry 3 of 5 File: PGPB Nov 29, 2001

DOCUMENT-IDENTIFIER: US 20010047471 A1
TITLE: System, method and article of manufacture to remotely configure and utilize an emulated device controller via an encrypted validation communication protocol

Cross Reference to Related Applications Paragraph (1):
[0001] This application is a continuation-in-part of (a) U.S. patent application Ser. No. 09/239,425 entitled "A Secure Electronic Mail System" filed on Jan. 28, 1999 and (b) Ser. No. 09/255,837 entitled "Method For Information Encoding And Transfer" filed on Feb. 23, 1999 which are continuation-in-part applications of co-pending U.S. patent application Ser. No. 08/892,982, filed Jul. 15, 1997, and entitled "Combined Remote Access and Security System"; which is a continuation-in-part of U.S. patent application Ser. No. 08/752,249, filed Nov. 19, 1996, and entitled "Combined Remote Access and Security System".

Summary of Invention Paragraph (6):
[0004] The present invention provides for secured, real-time, configuration and utilization of an emulated input/output device controller. The instant invention advances the art by allowing its practice to be supported via an encrypted communications protocol interfacing with, and relying upon, the teachings, practices and claims disclosed in co-pending U.S. patent applications Ser. No. 09/239,425 and 09/255,837 (hereinafter synonymously referred to as "Secure Agent" or "SA").

Summary of Invention Paragraph (7):
[0005] Secure Agent Service Overview

Summary of Invention Paragraph (8):
[0006] The following overview is provided to facilitate a comprehensive understanding of the teachings of the instant invention. Secure Agent utilizes a secure login sequence wherein a client connects to a Secure Agent server using a key known to both systems and a client connects and presents the server with user identification (as used herein the term "client" refers synonymously to a remote user establishing, and communicating with the instant invention through Secure Agent allocation and encryption processes as taught in the above noted applications). If recognized, the Secure Agent server initiates a protocol whereby the client's identification is verified and subsequent communication is conducted within a secured (encrypted) construct. For purposes of this overview, the term "server" should be considered a hardware configuration represented as a central processing unit wherein Secure Agent, a Host DLL and driver reside, and are executed. The term "DLL" as used herein refers to a Secure Agent host dynamically linked library (a.k.a. Host DLL). The term "DLL" or "dynamically linked library" is used in a manner consistent with that known to those skilled in the art. Specifically, the term "DLL" refers to a library of executable functions or data that can be used by a Windows application. As such, the instant invention provides for one or more particular functions and program access to such functions by creating a static or dynamic link to the DLL of reference, with "static links" remaining constant during program execution and "dynamic links" created by the program as needed.

Summary of Invention Paragraph (9):
[0007] The Secure Agent server presents a variable unit of data, such as the time of day, to the client as a challenge. The client must then encrypt that data and supply it back to the server. If the server is able to decrypt the data using the stored client's key so that the result matches the original unencrypted challenge data, the user is considered authenticated and the connection continue. The key is never passed between the two systems and is therefore never at risk of exposure.

Summary of Invention Paragraph (11):
[0009] The access rights of each client (what the client is able to accomplish during a session) is governed by data stored on the Secure Agent server to which the client is

associated. As an example, such rights might encompass the ability to administer and utilize the services of the server system, which would, in turn, include capabilities such as adding new client users, changing a user's rights, transferring new code to the server, using a feature (or service) of the server and more.

Summary of Invention Paragraph (12):
[0010] Consequently, Secure Agent allows for the transmission of new code to the server and for that code to be implemented upon demand by a client. Such dynamic, real-time implementation in turn, allows for the behavior of the server to be modified. It is to this behavior modification the instant invention addresses its teachings, and thereby advances the contemporary art.

Summary of Invention Paragraph (13):
[0011] As will be readily appreciated by those skilled in the art, though the instant invention utilizes encryption/decryption and code recognition technology associated with Secure Agent, an alternative technology may be employed in support of the instant invention without departing from the disclosure, teachings and claims presented herein.

Summary of Invention Paragraph (15):
[0012] The present invention is best viewed as comprised of two server components with one or more client subcomponents or sub-processes disclosed in association thereto. It can be further conceptualized that a distinguishable client component exists for each emulated device type recognized by the invention's server, with an individual client supporting the simultaneous use of a plurality of client-side components. As used throughout the instant invention specification and claims, the term "server" is used synonymously with "emulated device controller", "server central processing unit", "server CPU", and "remotely configurable input/output device controller" and the term "client" is used synonymously with "host user", "client central processing unit", "client CPU" and "remote user".

Summary of Invention Paragraph (16):
[0013] The invention's lower-most server component layer is a device driver to communicate directly with one or more hardware components attached to one or more computer systems, such as, but not limited to, mainframe computers (a.k.a. host processors). The driver controls the hardware in a manner prescribed by its design, causing it to interact with the other computer systems to which it is connected as if it were one or more device types (emulation). The driver additionally acts as a conduit to a higher level server component that governs the overall behavior of the emulated devices. This higher level component primarily supplies the driver with new data to provide through the emulated devices to the other computers to which it is connected and accepts data arriving to the emulated devices carried up by the device driver. Both layers predomoninantly operate on a device by device basis. The higher level server component, in turn, serves as the interface between Secure Agent technology and remotely connected clients allowing for the encrypted transmission of all data external to the server.

Summary of Invention Paragraph (18):
[0015] Just as a client might have the ability to administer users (i.e. add/remove), a client might be able to modify the presence and behavior of emulated devices, via Secure Agent administrative functions as taught by the afore noted pending patent applications. Allowable configuration ranges and values are verified and enforced according to rules by the server. The various data elements that may be controlled are listed at the bottom of this section. The server disallows modification of the active configuration (apart from device names and their security groups) and forces such modifications to be made to an inactive configuration. This inactive configuration may be swapped with the active configuration (thus activating it) upon demand. Thus, a new configuration may be prepared prior to a decision made to put it into effect. Additional control functionality includes but is not limited to the following:

Detail Description Paragraph (4):
[0047] In FIG. 1, a server CPU 103 has executing under control of its control program, Secure Agent software 106. The present invention advances the art and improves upon technology taught and claimed in the above noted pending applications, said applications and teachings incorporated by reference herein. The server 103 also has operating under control of its control program the remote configuration software 109 of the instant invention. Embodied within the server 103 is a hardware adaptor card 112. Said adaptor card 112 is in turn communicably attached to one or more host processors (121, 124, 127, 128). As used herein, the term "adaptor" refers synonymously to those

hardware configurations such as, but not limited to, "adaptor cards" which allow for connectability between two or more central processing units and the transference of data associated therewith. Illustrative non-limiting examples of such adaptors as used herein would include Crossroads ESCON adaptors, Crossroads ESCON parallel adaptors, Bus-Tech adaptors and IBM ESCON adaptors.

Detail Description Paragraph (6):
[0049] Lastly shown in FIG. 1 is a Security Administrator client 151 interactively communicating with the Secure Agent software 106 operating within the server 103. As will be discussed in further detail and in association with FIGS. 2 through 7, the Security Administrator 151 utilizes Secure Agent software 106 to administer and maintain user/resource profiles 157 and further communicates with information conveyed to said Secure Agent software 106 via the software processes associated with the remote configuration software 109 of the instant invention.

Detail Description Paragraph (8):
[0051] The following discussion in association with FIG. 1 provides a brief non-limiting synopsis of the teachings of the instant invention and generally discusses the interrelationships of hardware and software processing components of the instant invention. In FIG. 1, a Security Administrator 151 defines via Secure Agent software 106, user and resource profiles 157. Such profiles are stored in a non-volatile storage medium, such as but not limited to, a disk drive 158. User resource records are those records which typically define security group or groups, and access control variables associated with the user. Stated succinctly, the user resource record/profile defines those resources that the user may utilize and the bounds of such utilization. The Security Administrator 151 may also define resource profiles, such resource profiles define the device type and grouping of emulated input/output devices as well as central processing unit designations associated with each emulated device type and/or grouping. When attempting to establish a session between a host user 145 and any one of the operating systems and/or application programs operating under the dispatching control of the operating systems of host processors 128, 121, 124 or 127, a user via a communications network 148 communicates first with Secure Agent software 106 operating within the server 103 of the instant invention 109. Assuming the user 145 is recognized as an authenticated and authorized user of the system as governed by Secure Agent software 106, the user 145 next requests a device or a device grouping of emulated input/output devices he or she anticipates utilizing in the requested session. The Secure Agent software 106 verifies the user 145 as authority to allocate such emulated input/output devices and correspondingly associates such devices with the user and user session between one or more of the host processors 128, 121, 124 and 127. Once established, the session continues as normal with input/output requests of the user serviced via emulated input/output device as opposed to the real input/output devices 131 associated with one or more of the host processors. Upon completion of the session or a specific deallocation request initiated by the user, the client termination subprocess of the instant invention deallocates the emulated input/output device or devices. As indicated, the processing subcomponents of the instant invention further include Adaptor Configuration Load, Client Communication, Client Termination, Administration, Server Initialization and Server Termination subprocesses. It is to such subprocesses FIGS. 2 through 7 address themselves. A more detailed disclosure of each subprocess follows.

Detail Description Paragraph (12):
[0055] Subsequent to the driver initialization, the Host DLL initializes variables it utilizes 211 and clears a user connection block to allow information for each user to be represented 212. The Host DLL further exposes and makes available to Secure Agent a block of data, representing an emulated device specific administrative instruction set 213, for each user. In addition to such normal data elements as a user ID and password, this instruction set advises Secure Agent to maintain device type and security group strings on behalf of each user specifically for the support of this Host DLL. The device types limits those types of emulated devices to which a user might claim access whereas the device security groups name the emulated device security groups to which a user is subscribed. In addition, at this stage linkage to configuration support routines within the Host DLL is also established. As practiced in one embodiment of the invention, the root name of the administrative tree structure is exposed to Secure Agent indicating that the Host DLL supports the configuration of information and will respond in a positive manner to requests for information and management of branches under this particular root. The Host DLL next creates a mutex serialization mechanism to be used by configuration support routines during access of adaptor configuration data to insure data integrity 214. This serialization mechanism is used to prevent for example potential simultaneous updates by multiple administrators as well as to prevent

a client from enumerating emulated devices while it is being manipulated.

Detail Description Paragraph (13):
[0056] The Host DLL continues to open or otherwise establishes communication with the driver 215 and requests from it a number of recognized adaptors 216 to which the driver responds 217, whereupon the Host DLL requests from the driver its version number 218 to which the driver also responds 219. The Host DLL then records into a Secure Agent log the driver version and the number of adaptors it controls 220, and proceeds to indicate that each adaptor is not yet in a condition to support client connectivity 221. Data representing the adaptor configuration to be utilized (the active configuration) is next loaded 223. This data specifies device types and number of devices to be emulated, in conjunction with user-friendly (readable) names and security groups for each such emulated device. A second unique set of this data is loaded (the inactive configuration) 224 on behalf of this same adaptor to be used as a work area for administrators. This allows administrators to accumulate a series of configuration changes prior to effecting the activation of those changes as a whole. During said initialization, the Host DLL lastly ensures that the loaded adaptor configurations are within operationally permissible parameters 225.

Detail Description Paragraph (17):
[0060] In FIG. 4 the Host DLL first indicates the adaptor's unavailability 401 and for each client currently connected to a logical unit on this adaptor, issues a message to the client indicating that the client is being disconnected due to administrative device management 402. The Host DLL then performs the client disconnection services in association with the invention's Client disconnection subprocess as will be discussed in further detail in association with FIG. 6. The Host DLL continues by next recording into Secure Agent log the configuration for this adaptor is being loaded 403 and if the adaptor is to be forced offline to the mainframe to which it is connected 404, prepare and uses an empty configuration indicating that Emulated devices are not to be emulated during this session. If the adaptor is not to be forced offline, an active configuration for the adaptor is provided and a request that the adaptor using the active configuration data is initiated 405. The driver as instructed causes the adaptor to be offline to the channel at this stage in the adaptor configuration load 406, destroys each of the adaptor emulated devices driver object instances 407 causing or eliminating the exposure of emulated devices support to applications through NT 408 and frees all allocated storage and resources 409. The driver next determines if Emulated devices are to be emulated 410 and then request that the adaptor be brought online to the channel 411, lastly indicating that the adaptor is available for client use 412.

Detail Description Paragraph (20):
[0063] In FIG. 5, a client connection first initializes variables that it utilizes 501 then employs Secure Agent client code in order to establish a connection to the Host DLL 502, whereupon the Host DLL retains the client's name 503 and loads the client's device type and security groups 504. A new client object instance is then created to represent this new client connection with the variables it will use becoming initialized 505. The Host DLL then stores the location of the client object in a user connection block 506. At this point the client sends to the Host DLL the command version level that represents the client feature set as a means to facilitate backward compatibility by future Host DLLs 507 which the Host DLL stores for possible reference 508. By knowing the version of the client, the Host DLL can and will prevent communicating with older clients in a manner supported only by newer clients, whereas newer clients will be able to take advantage of a fuller set of features that the Host DLL offers. The client next provides to the Host DLL the emulated device type in which it is interested 509 whereupon the Host DLL stores it for later reference 510. The client then requests of the Host DLL its command version level 511 that the client stores for possible reference 513. Just as with the Host DLL being able to restrict its behavior for older clients, since the client knows the version level of the Host DLL it can restrict itself from attempting to take advantage of features available only on newer servers whereas newer servers might be more fully exploited. The client then requests from the Host DLL a list of the currently available emulated devices to which the client may connect 514. The Host DLL returns the response back 515 whereupon the client selects one of the emulated devices and requests that the Host DLL establish a connection to it on its behalf 516.

CLAIMS:

1. A system to facilitate remote configuration and utilization of an emulated device controller via communication of encrypted data external to said controller comprising: at least one security administrator central processing unit communicably attached to an

emulation server central processing unit; at least one client central processing unit communicably attached to said emulation server central processing unit; first software means for validating authorized remote user access and encryption of data, said first software executing from within said emulation server central processing unit; second software means for facilitating remote configuration and utilization of an emulated device controller; said second software executing from within said emulation server central processing unit; at least one hardware adaptor card communicably attached to said emulation server central processing unit; and at least one host processor central processing unit communicably attached to said emulation server central processing unit via said hardware adaptor card.

WEST

Generate Collection

L13: Entry 4 of 5 File: PGPB Oct 25, 2001

DOCUMENT-IDENTIFIER: US 20010034659 A1
TITLE: Simplified method and system for e-commerce operable in on-line and off -line modes

Summary of Invention Paragraph (33):
[0032] (f) the ability to initiate or perform Internet transactions for storage or later processing, when the end user's remote access device is off-line or otherwise not able to be in instantaneous real-time contact with the Internet.

Detail Description Paragraph (34):
[0073] With further reference to FIG. 4, the e-navigator and e-agent relationship is that the e-navigator 60 is a user interface which helps customers post their demands and needs into a secure virtual private e-agent. The e-agent 30 is the "behind the scene" manager, which saves customer information, purchase history, and provides e-solutions such as e-search and match to match customers demands with supplier information; as well as e-care (product and customer care information); e-billing (EBPPP, e-history; and e-logistics (delivery information).

Detail Description Paragraph (35):
[0074] By opening an application program interface (API) between the e-navigator and the e-agent, anyone can design any "plug-in" customized e-navigators. E-scan is one of several applications which are specialized for mobile and remote users.

WEST

Generate Collection

L21: Entry 1 of 3 File: USPT Mar 18, 2003

DOCUMENT-IDENTIFIER: US 6535743 B1
TITLE: System and method for providing directions using a communication network

Detailed Description Text (13):
Modem 44 may comprise any suitable remote access device that supports communication with NSC 14 using any suitable communication protocol. In one embodiment, modem 44 supports simultaneous voice and data communications over voice network 18 and/or global computer network 98.

Detailed Description Text (17):
Power controller 52 comprises a power supply and associated power control circuitry. The power supply provides a constant supply voltage from the battery associated with vehicle 25, and an internal battery for critical data retention when no vehicle battery is present. The power control circuitry provides input power filtering, limiting, protection, monitoring, switching, and control to enable mobile unit 12 to remain powered up even if the ignition associated with vehicle 25 is lost. Upon the loss of ignition, platform 24 will wait until a configurable power down delay, stored in configuration data 80, has elapsed and all processing is complete, including any calls in progress, prior to powering itself down. In one embodiment, power controller 52 supports the ability to place transceiver 42 in a low power state even when mobile unit 12 is powered down. In the low power state, transceiver 42 simply monitors the overhead channel for pages and orders. Upon the receipt of a page directed to one of the supported NAMs, transceiver 42 wakes up mobile unit 12, and responds to the page.

Detailed Description Text (22):
Computing device 30 comprises a hand-held or laptop computer, a personal digital assistant (PDA), a personal information manager (PIM), or any other portable computing device. Device 30 exchanges information with the various components of mobile unit 12. In particular, device 30 interfaces with processor 38 using bus 32 and appropriate interfacing software to download and upload data regarding user interface 22, platform 24, sensors 26, actuators 28, or any other component of mobile unit 12. In one embodiment, device 30 uses a network interface 94 to maintain an external link 96 that provides communication capabilities between mobile unit 12 and a global computer network 98, such as the Internet. Network interface 94 comprises a modem, a remote access device, or any other suitable collection of hardware and/or software to provide communication capability between device 30 and network 98. By interfacing computing device 30 with processor 38, mobile unit 12 facilitates an exchange of information with an external and portable source.

Detailed Description Text (24):
Mobile units 12 couple to NSC 14 using voice network 18. Voice network 18 comprises cell transmitter sites 100, mobile switching centers (MSCs) 102, and the various components of the public switched telephone network (PSTN) 104. Voice network 18 may also include any other suitable land-based or satellite-based transmitting and receiving components. Voice network 18 generally comprises any suitable number and collection of telecommunication hardware and associated software that provides a voice path between mobile unit 12 and NSC 14. In one embodiment, voice network 18 comprises a first voice network 106 that supports traditional voice services, such as, for example, sending and receiving voice calls, and a second voice network 108 that supports enhanced services provided by service centers 16. Networks 106 and 108 are depicted as separate components in FIG. 1 for convenience and illustrative purposes only, and it should be understood that the present invention contemplates networks 106 and 108 as the same or different networks.

Detailed Description Text (25):
Voice paths 110 couple voice network 18 to NSC 14. Voice paths 110 may be individual lines, multiple lines, trunks, multiple trunks, or any other suitable collection of

lines, trunks, and/or other suitable paths to support one or more voice paths. In one embodiment, incoming voice path 112 establishes voice communication between mobile unit 12 and NSC 14, and outgoing voice path 114 establishes communication between NSC 14 and service center 16. For example, NSC 14 may couple to service center 16 using a voice path that includes outgoing voice path 114, PSTN 104, and voice path 116. Although FIG. 1 illustrates NSC 14 coupled to a service center 16 using PSTN 104 and voice paths 114 and 116, it should be understood that NSC 14 may couple to service centers 16 using a dedicated voice path separate from PSTN 104. In a particular embodiment, NSC 14 may also couple to service center 16 using a data path that includes data path 118, data network 20, and data path 120.

Detailed Description Text (27):
Data network 20 may include hardware and software to establish a dedicated data path between NSC 14 and service center 16, using frame relay, X.25, TCP/IP, or any other suitable dedicated communication protocol. Alternatively, data network 20 may include hardware and software to implement a non-dedicated, switched, or dial-up data path between NSC 14 and service center 16. Data network 20 and data paths 118 and 120 may be wireline, wireless, or a combination of wireline and wireless technologies. For example, data network 20 may comprise a portion of PSTN 104 that establishes a dial-up modem connection that is separate from voice path 114. In a particular embodiment, the data path established by data network 20 and data paths 118 and 120 provide a sufficiently small transmission time to enable data associated with the communication session to arrive at service center 16 simultaneously or in advance of the call over the voice path established by PSTN 104 and voice paths 114 and 116.

Detailed Description Text (40):
NSC 14 then establishes a voice path (e.g. by initiating a voice call) with the selected service center 16 using PSTN 104 and voice paths 114 and 116 or, optionally, using a dedicated voice path separate from PSTN 104. In one embodiment, NSC 14 also communicates a data message to service center 16 using data network 20 and data paths 118 and 120. In another embodiment, NSC 14 communicates a data message to service center 16 using PSTN 104, and voice paths 114 and 116 using modems, DTMF techniques, and/or out-of-band signaling. For example, NSC 14 may forward to service center 16 a data message containing the history and specifications of mobile unit 12, the medical history of the occupants of mobile unit 12, the information provided by service message 58, and any other suitable information. Both the voice call and the data message from NSC 14 to service center 16 may include an identifier of mobile unit 12.